Unidad 6
Entrada- Salida

Índice

Gestión de Entrada/Salida **4**

Gestión de E/S. Planificación de Discos **5**
- Dispositivos de E/S 5
  - Dispositivos internos 5
  - Dispositivos externos 5
  - Objetivos en el Diseño de E/S 5
- Técnicas para la Organización de la Función de E/S 6
- Gestión de E/S por muestreo o E/S programada 6
- Gestión de E/S por interrupciones 6
- Gestión de E/S híbrida 7
- Canales de E/S 7
- Principios del Hardware de E / S 8
- Dispositivos de E/S 8
- Controladores de Dispositivos 8
- Acceso Directo a Memoria (DMA) 10
- Principios del Software de E / S 12
- Objetivos del Software de E / S 12
  - Manejadores de Interrupciones 13
  - Manejadores de Dispositivos 13
  - Software de E / S Independiente del Dispositivo 14
  - Software de E / S en el Espacio del Usuario 14

Discos - Hardware Para Discos **15**
- Discos 15
- Hardware Para Discos 15
- Operación de Almacenamiento de Disco de Cabeza Móvil 15
- Algoritmos de Programación del Brazo del Disco 17
- Porqué es Necesaria la Planificación de Discos 18
- Características Deseables de las Políticas de Planificación de Discos 19
- Optimización de la Búsqueda en Discos 19
  - Planificación FCFS (Primero en Llegar, Primero en Ser Servido) 20
  - Planificación SSTF (Menor Tiempo de Búsqueda Primero) 20
  - Planificación SCAN 20
  - Planificación SCAN de N - Pasos 20
  - Planificación C - SCAN (Búsqueda Circular) 20
  - Esquema Eschenbach 20
  - Conclusiones 21
- Optimización Rotacional en Discos 21
- Consideraciones de los Discos Sobre los Sistemas 21
- Manejo de Errores en Discos 22
- Ocultamiento de Una Pista a la Vez en Discos 22
- Discos en RAM 23

Relojes 23
Terminales 24
Administración de Dispositivos 26
Medios de Almacenamiento de Acceso Secuencial. 27
Medios de Almacenamiento de Acceso Directo (DASD). 27
DASD de Cabeza Fija. 28
DASD de Cabeza Móvil. 28
En Dispositivos de cabeza fija. 29
En Dispositivos de cabeza móvil. 29
Componentes del subsistema de entrada/salida. 30
Comunicación entre dispositivos. 30
Administración de las solicitudes de entrada/salida. 31
Estrategias de búsqueda de manejo de dispositivos. 32
RAID 32

El sistema operativo: gestión de entrada/salida 36
Introducción y evolución histórica 36
Esquemas hardware de gestión de E/S 37
Polling 37
Interrupciones 37
Acceso directo a memoria (DMA) 38

Gestión de la E/S por parte del SO 38
Tipos de dispositivos 39
Dispositivos de bloques y caracteres 39
Dispositivos de red 39
Mejoras del rendimiento 40
Buffering 40
Políticas de planificación de discos 40
Caché de disco 41
Sincronización con el disco 41
Políticas de reemplazo 42

Gestión de dispositivos 43
Introducción 43
Características de los dispositivos 43
Tipos de entrada/salida 44

Modelo general de la entrada/salida 44
Gestión de la entrada/salida por capas 44
Esquema cliente-servidor 46
Representación de la E/S 46
Rutinas de E/S 47
Manejadores de dispositivos 48
Almacenamiento intermedio de la E/S 49

Gestión de discos **50**
Arranque y parada del motor 51
Traducción de bloques 51
Reubicación de sectores. Interleaving de sectores. 51
Planificación de accesos 52
Tratamiento de errores 53
Evaluación del rendimiento 54
Esquema de un manejador de discos 54
La aplicación y la I/O 56

Diferentes características que tienen los dispositivos **57**

Dispositivos de bloque y de carácter **57**

Memory mapped I/O **58**

Network devices. Sockets **58**
I/O bloqueante y no bloqueante 59
El subsistema de I/O del kernel 59

Entrada- salida: estructuras en memoria secundaria **59**
Estructura de disco 59
Distintos tipos de algoritmos de scheduling 60
FCFS 60
SSTF 60
SCAN 61
C-SCAN 61
LOOK 61

Cómo se selecciona el algoritmo a utilizar **61**
Otras características de administración de disco 62

Formateo de disco **62**

Boot Block **64**

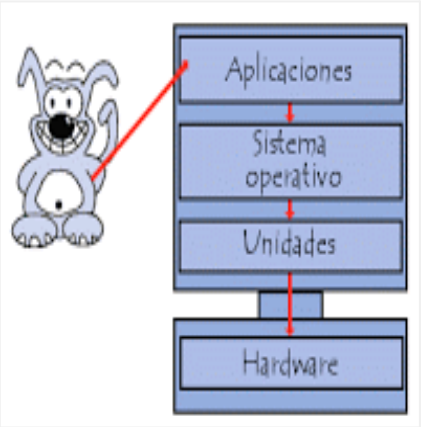
Gestión de Entrada/Salida

Una de las funciones principales de un S. O. es el control de todos los dispositivos de E/S de la computadora.
Las principales funciones relacionadas son:

- Enviar comandos a los dispositivos.
- Detectar las interrupciones.
- Controlar los errores.
- Proporcionar una interfaz entre los dispositivos y el resto del sistema:
    - Debe ser sencilla y fácil de usar.
    - Debe ser la misma (preferentemente) para todos los dispositivos (independencia del dispositivo).

El código de e / s representa una fracción significativa del S. O.
El uso inapropiado de los dispositivos de e / s frecuentemente genera ineficiencias del sistema, lo que afecta la performance global.

Consiste en un sistema de almacenamiento temporal (caché), una interfaz de manejadores de dispositivos y otra para dispositivos concretos. El sistema operativo debe gestionar el almacenamiento temporal de E/S y servir las interrupciones de los dispositivos de E/S.
Entrada y salida designa cualquier transferencia de información desde o hacia memoria o el procesador. Comprende tanto la transferencia entre diversos niveles de la memoria como la comunicación con los periféricos. El sistema de entrada y salida es la parte del S.O. encargada de la administración de los dispositivos de e/s.
Este sistema proporciona un medio para tratar los archivos y dispositivos de manera uniforme, actuando como interfaz (debe ser independiente, sencilla y fácil de utilizar) entre los usuarios y los dispositivos de e/s que pueden ser manipulados por órdenes de alto nivel. Entrada y salida designa cualquier transferencia de información desde o hacia memoria o el procesador.
Comprende tanto la transferencia entre diversos niveles de la memoria como la comunicación con los periféricos.



Gestión de E/S. Planificación de Discos

Dispositivos de E/S

Dispositivos internos

Los principales dispositivos internos son los discos RAM, que usan una porción de memoria pre-asignada para almacenar los bloques. Tienen la ventaja de que el acceso es instantáneo (a la velocidad de la memoria central).

Dispositivos externos

Se clasifican en:

- **Leíbles por humanos:** apropiados para comunicarse con el usuario. Ej: Mouse, terminales de video, etc.
- **Leíbles por la máquina:** para comunicarse con el equipo electrónico. Ej; discos, sensores, drivers de cinta, etc.
- **Comunicación:** para comunicarse con drivers remotos. Ej: Líneas digitales, modems, etc.

Existen grandes diferencias entre estas clases de dispositivos de E/S. Las principales son:

- **Velocidad de transmisión de datos.**
- **Software, Hardware y políticas de apoyo de S.O. que requiere el dispositivo.**
- **Complejidad de control:** Se refiere a la complejidad que requieren los dispositivos de I/O, Una impresora necesita una interface de control más simple que un disco.
- **Unidad de transferencia:** puede ser como una cadena de bytes o caracteres o como largos bloques.
- **Representación de los datos:** Cada dispositivo utiliza distintos códigos de datos de programa, incluyendo diferentes código de caracteres y conversiones de paridad.
- Condiciones de error: Cada dispositivo difiere en naturaleza de error, como se reportan, las consecuencias, etc.

Objetivos en el Diseño de E/S

- **Eficiencia:** la mayoría de los dispositivos de I/O son extremadamente lentos comparados con la memoria principal y el procesador. Por esto se necesita la multiprogramación. Permite que algunos procesos esperen en las operaciones de I/O mientras otro se ejecuta. Sin embargo, se sigue malgastando tiempo de procesador. Para esto se puede utilizar Swapping, que trae

procesos listos adicionales para mantener al procesador ocupado. Finalmente, el mejor esfuerzo de programa de diseño para mejorar la eficiencia de I/O fue el propio disco de I/O.

- **Generalidad:** se trata de manejar un número de dispositivos de manera uniforme. Como es difícil alcanzar generalidad entre las distintas características de los dispositivos, se utiliza un mecanismo para el diseño de los dispositivos de I/O que se encarga de esconder la mayoría de los detalles de los dispositivos en el nivel más bajo de rutinas para que los procesos y los niveles mas altos del SO vean a los dispositivos como funciones generales. Ej.: leer, cerrar, abrir, etc

Técnicas para la Organización de la Función de E/S

El sistema operativo puede gobernar el dispositivo E/S a través del gestor de dispositivos de tres maneras posibles:

- Gestión de E/S por muestreo.
- Gestión de E/S por interrupciones.
- Gestión híbrida de E/S.

Gestión de E/S por muestreo o E/S programada

En el procesador se activa un comando de I/O, en representación de un proceso y hacia un modulo de I/O, luego ese proceso tiene tiempos de espera hasta que la operación se complete para proceder. Ejecuta cuatro tipos de comandos: control, verificación, lectura y escritura.
El gestor de dispositivo periódicamente comprueba el estado del dispositivo. Para ello comprueba mediante los registros de estado del puerto de E/S si hay datos provenientes del dispositivo que deben ser tratados. Un ejemplo de lectura de un único dispositivo sería:

```
envia_orden_a_dispositivo(...);
while (!orden_terminada());
        procesar_dato_resultado_de_orden();
```

La "lista de dispositivos" se puede recorrer en base a dos criterios distintos:

- **Con prioridad uniforme:** Todos los dispositivos reciben las mismas oportunidades de ser atendidos: se pregunta al primer dispositivo si requiere atención, y se le atiende. Después se pasa al siguiente, y así sucesivamente hasta dar una vuelta completa.
- **Con prioridad escalonada:** Tras atender a cualquier dispositivo, se vuelve a comenzar el sondeo por el primer dispositivo. El orden es determinante, pues se establece una relación de prioridad entre dispositivos determinada por ese orden.

Es ineficaz en sistemas interactivos, debido a que requiere espera ocupada y resulta engorrosa para atender a varios dispositivos simultáneamente, al implicar establecer prioridades entre estos.

Gestión de E/S por interrupciones

Es igual que el anterior pero antes de terminar la operación o las instrucciones, el módulo de I/O envía una señal de interrupción. Se suspende el proceso y se prepara otro trabajo. Esto incrementa la eficiencia.

Una interrupción por hardware es una señal proveniente de un dispositivo de E/S para notificar al procesador de un cierto evento que debe ser tratado. Por ejemplo, un proceso hace uso de la llamada al sistema read para solicitar información que se encuentra en disco magnético. Una vez que la información está disponible, la interrupción es empleada para que se ejecute el gestor de dispositivos y obtenga los datos para dicho proceso, que ya están disponibles.
Las interrupciones son un mecanismo que ofrece la arquitectura para conectar los dispositivos de E/S con el procesador. No obstante, los datos provenientes del dispositivo de E/S se obtienen a través del bus.
El dispositivo de E/S emplea una de las líneas de interrupción que conectan al dispositivo con el procesador. Cada una de estas líneas corresponden con un cierto dispositivo o una familia de dispositivos de naturaleza similar. La asignación de líneas y dispositivo es estática y sucede en tiempo de arranque, por tanto, no cambia a lo largo del tiempo. En caso de que haya datos a tratar en el dispositivo, se notifica al procesador mediante la línea de interrupción. Ante esto, el planificador debe apartar el proceso que esté en estado activo para conmutar al gestor de dispositivo, el cual realizará el tratamiento de la interrupción, obteniendo los datos del dispositivo de E/S que están pendientes de ser tratados.
Por tanto, por cada interrupción se debe conmutar al gestor de dispositivos.
Las interrupciones no son reentrantes, esto quiere decir que cuando se está realizando el tratamiento de una interrupción se desactiva temporalmente la notificación por interrupciones. Por tanto, una interrupción se ejecuta hasta fin de tratamiento, y en ningún caso es interrumpida por otra interrupción.

Gestión de E/S híbrida

Esta aproximación es la más usada en sistemas operativos modernos, y consiste en emplear una combinación de las soluciones anteriores. En principio, se realiza una gestión por interrupciones, pero ante situaciones de estrés en las que se ofrezca una carga de trabajo muy alta, que pueda llevar a una sobrecarga de conmutaciones, se limita en el tiempo la consulta de datos. Por ejemplo, si la tarjeta de red estuviera saturando el procesador con interrupciones, se puede optar por atenderlas todas juntas periódicamente (En lotes).
Ejemplo: **Acceso directo a memoria (DMA):** Controla el intercambio de datos entre la memoria principal y el módulo de I/O. El procesador envía una petición de transferencia de un bloque de datos a la DMA y se interrumpe sólo cuando todo el bloque es transferido. El procesador solo se involucra al principio y al final del proceso.

Canales de E/S

El canal de E/S es una extensión del concepto de DMA. Un canal de E/S tiene la capacidad de ejecutar instrucciones de E/S, lo que le da un control total sobre las operaciones de E/S. En un sistema informático que conste de tales dispositivos, las instrucciones de E/S se almacenan en la memoria principal y serán ejecutadas por un procesador de propósito específico en el mismo canal de E/S. Así, la CPU inicia una transferencia de E/S ordenando al canal que ejecute un programa en la memoria. Los canales de E/S pueden realizar las transferencias de datos en serie o en paralelo.

Hay dos tipos comunes de canales de E/S:

- **Canal selector:** controla varios dispositivos y transfiere datos de estos dispositivos, uno por vez.
- **Canal multiplexor:** puede manejar la E/S con varios dispositivos al mismo tiempo.

Principios del Hardware de E / S

El enfoque que se considerará tiene que ver con la interfaz que desde el hardware se presenta al software:

- Comandos que acepta el hardware.
- Funciones que realiza.
- Errores que puede informar.

Dispositivos de E/S

Se pueden clasificar en dos grandes categorías:

- Dispositivos de bloque.
- Dispositivos de carácter.

Las principales características de los dispositivos de bloque son:

- La información se almacena en bloques de tamaño fijo.
- Cada bloque tiene su propia dirección.
- Los tamaños más comunes de los bloques van desde los 128 bytes hasta los 1.024 bytes.
- Se puede leer o escribir en un bloque de forma independiente de los demás, en cualquier momento.
- Un ejemplo típico de dispositivos de bloque son los discos.

Las principales características de los dispositivos de caracter son:

- La información se transfiere como un flujo de caracteres, sin sujetarse a una estructura de bloques.
- No se pueden utilizar direcciones.
- No tienen una operación de búsqueda.
- Un ejemplo típico de dispositivos de caracter son las impresoras de línea, terminales, interfaces de una red, ratones, etc.

Algunos dispositivos no se ajustan a este esquema de clasificación, por ejemplo los relojes, que no tienen direcciones por medio de bloques y no generan o aceptan flujos de caracteres.
El sistema de archivos solo trabaja con dispositivos de bloque abstractos, por lo que encarga la parte dependiente del dispositivo a un software de menor nivel, el software manejador del dispositivo.

Controladores de Dispositivos

Las unidades de E/S generalmente constan de:

- Un componente mecánico.
- Un componente electrónico, el controlador del dispositivo o adaptador.

Muchos controladores pueden manejar más de un dispositivo.
El S.O. generalmente trabaja con el controlador y no con el dispositivo.
Los modelos más frecuentes de comunicación entre la CPU y los controladores son:

- Para la mayoría de las micro y mini computadoras:
    - Modelo de bus del sistema.

- Para la mayoría de los mainframes:
    - Modelo de varios buses y computadoras especializadas en e / s llamadas canales de e / s.

La interfaz entre el controlador y el dispositivo es con frecuencia de muy bajo nivel:

- La comunicación es mediante un flujo de bits en serie que:
    - Comienza con un preámbulo.
    - Sigue con una serie de bits (de un sector de disco, por ej.).
    - Concluye con una suma para verificación o un código corrector de errores.
- El preámbulo:
    - Se escribe al dar formato al disco.
    - Contiene el número de cilindro y sector, el tamaño de sector y otros datos similares.

El controlador debe:

- Convertir el flujo de bits en serie en un bloque de bytes.
- Efectuar cualquier corrección de errores necesaria.
- Copiar el bloque en la memoria principal.

Cada controlador posee registros que utiliza para comunicarse con la cpu:

- Pueden ser parte del espacio normal de direcciones de la memoria: e / s mapeada a memoria.
- Pueden utilizar un espacio de direcciones especial para la e / s, asignando a cada controlador una parte de él.

El S. O. realiza la e / s al escribir comandos en los registros de los controladores; los parámetros de los comandos también se cargan en los registros de los controladores.

Al aceptar el comando, la cpu puede dejar al controlador y dedicarse a otro trabajo.

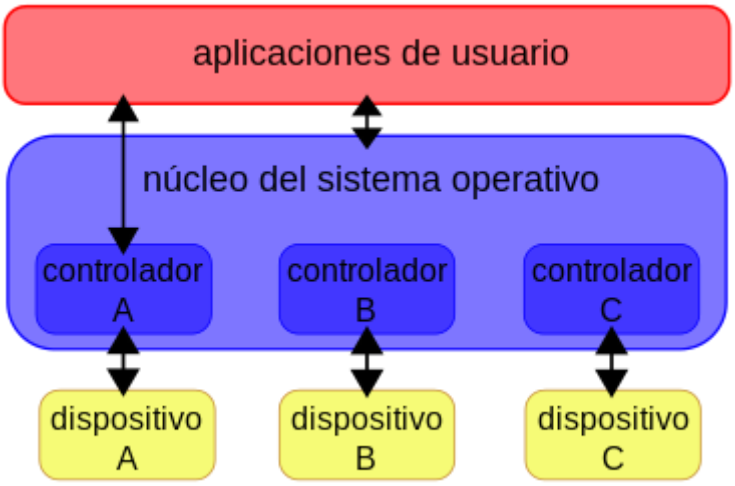
Al terminar el comando, el controlador provoca una interrupción para permitir que el S. O.:

- Obtenga el control de la cpu.
- Verifique los resultados de la operación.

La cpu obtiene los resultados y el estado del dispositivo al leer uno o más bytes de información de los registros del controlador.

Ejemplos de controladores, sus direcciones de e / s y sus vectores de interrupción en la PC IBM pueden verse en la Tabla 5.1.

Controlador de E/S	**Dirección de E/S**	**Vector de interrupciones**
Reloj	040 - 043	8
Teclado	060 - 063	9
Disco duro	320 - 32f	13
Impresora	378 - 37f	15
Disco flexible	3f0 - 3f7	14
Rs232 primario	3f8 - 3ff	12
Rs232 secundario	2f8 - 2ff	11



Acceso Directo a Memoria (DMA)

Muchos controladores, especialmente los correspondientes a dispositivos de bloque, permiten el DMA.
Si se lee el disco sin DMA:

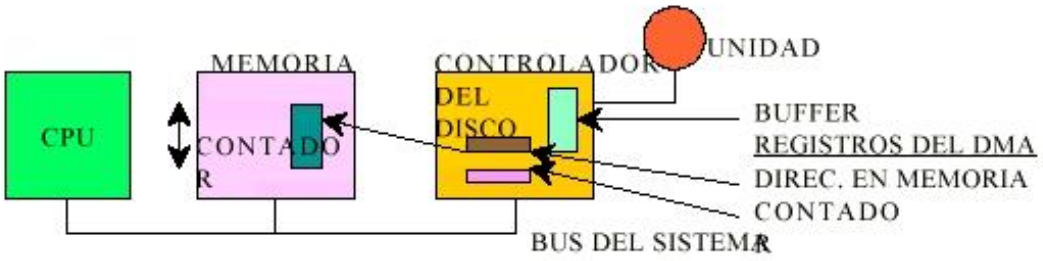
- El controlador lee en serie el bloque (uno o más sectores) de la unidad:
  - La lectura es bit por bit.
  - Los bits del bloque se graban en el buffer interno del controlador.
- Se calcula la suma de verificación para corroborar que no existen errores de lectura.
- El controlador provoca una interrupción.
- El S.O. lee el bloque del disco por medio del buffer del controlador:
  - La lectura es por byte o palabra a la vez.
  - En cada iteración de este ciclo se lee un byte o una palabra del registro del controlador y se almacena en memoria.
- Se desperdicia tiempo de la cpu.

DMA se ideó para liberar a la cpu de este trabajo de bajo nivel.
La cpu le proporciona al controlador:

- La dirección del bloque en el disco.
- La dirección en memoria adonde debe ir el bloque.
- El número de bytes por transferir.

Luego de que el controlador leyó todo el bloque del dispositivo a su buffer y de que corroboró la suma de verificación:

- Copia el primer byte o palabra a la memoria principal.
- Lo hace en la dirección especificada por medio de la dirección de memoria de DMA.
- Incrementa la dirección DMA y decrementa el contador DMA en el número de bytes que acaba de transferir.
- Se repite este proceso hasta que el contador se anula y por lo tanto el controlador provoca una interrupción.
- Al iniciar su ejecución el S. O. luego de la interrupción provocada, no debe copiar el bloque en la memoria, porque ya se encuentra ahí (ver Figura 5.1).



Figura˜5.1: Un controlador realiza completamente una transferencia DMA.

El controlador necesita un buffer interno porque una vez iniciada una transferencia del disco:

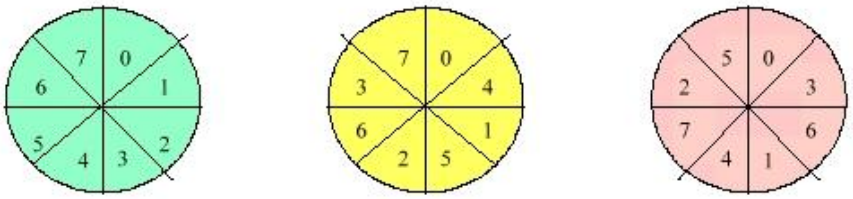
- Los bits siguen llegando del disco constantemente.
- No interesa si el controlador está listo o no para recibirlos.
- Si el controlador intentara escribir los datos en la memoria directamente:
  - Tendría que recurrir al bus del sistema para c/u de las palabras (o bytes) transferidas.
  - El bus podría estar ocupado por otro dispositivo y el controlador debería esperar.
  - Si la siguiente palabra llegará antes de que la anterior hubiera sido almacenada, el controlador la tendría que almacenar en alguna parte.

Si el bloque se guarda en un buffer interno:

- El bus no se necesita sino hasta que el DMA comienza.
- La transferencia DMA a la memoria ya no es un aspecto crítico del tiempo.

Los controladores simples no pueden atender la e / s simultánea:

- Mientras transfieren a la memoria, el sector que pasa debajo de la cabeza del disco se pierde; es decir que el bloque siguiente al recién leído se pierde.
- La lectura de una pista completa se hará en dos rotaciones completas, una para los bloques pares y otra para los impares.
- Si el tiempo necesario para una transferencia de un bloque del controlador a la memoria por medio del bus es mayor que el tiempo necesario para leer un bloque del disco:
  - Sería necesario leer un bloque y luego saltar dos o más bloques.
  - El salto de bloques:
    - Se ejecuta para darle tiempo al controlador para la transferencia de los datos a la memoria.
    - Se llama separación.
    - Al formatear el disco, los bloques se numeran tomando en cuenta el factor de separación (ver Figura 5.2).
    - Esto permite al S. O.:
      - Leer los bloques con numeración consecutiva.
      - Conservar la máxima velocidad posible del hardware.



Figura~5.2: Factores de separación: sin separación, separación simple y separación doble.

Principios del Software de E / S

La idea básica es organizar el software como una serie de capas donde:

- Las capas inferiores se encarguen de ocultar las peculiaridades del hardware a las capas superiores.
- Las capas superiores deben presentar una interfaz agradable, limpia y regular a los usuarios.

Objetivos del Software de E / S

Un concepto clave es la independencia del dispositivo:

- Debe ser posible escribir programas que se puedan utilizar con archivos en distintos dispositivos, sin tener que modificar los programas para cada tipo de dispositivo.
- El problema debe ser resuelto por el S. O.

El objetivo de lograr nombres uniformes está muy relacionado con el de independencia del dispositivo.
Todos los archivos y dispositivos adquieren direcciones de la misma forma, es decir mediante el nombre de su ruta de acceso.
Otro aspecto importante del software es el manejo de errores de e / s:

- Generalmente los errores deben manejarse lo más cerca posible del hardware.
- Solo si los niveles inferiores no pueden resolver el problema, se informa a los niveles superiores.
- Generalmente la recuperación se puede hacer en un nivel inferior y de forma transparente.

Otro aspecto clave son las transferencias síncronas (por bloques) o asíncronas (controlada por interruptores):

- La mayoría de la E/S es asíncrona: la cpu inicia la transferencia y realiza otras tareas hasta una interrupción.
- La programación es más fácil si la e / s es síncrona (por bloques): el programa se suspende automáticamente hasta que los datos estén disponibles en el buffer.

El S. O. se encarga de hacer que operaciones controladas por interruptores parezcan del tipo de bloques para el usuario.
También el S. O. debe administrar los dispositivos compartidos (ej.: discos) y los de uso exclusivo (ej.: impresoras).
Generalmente el software de e / s se estructura en capas (ver Figura 5.3):

- Manejadores de interrupciones.
- Directivas de dispositivos.
- Software de S. O. independiente de los dispositivos.
- Software a nivel usuario.

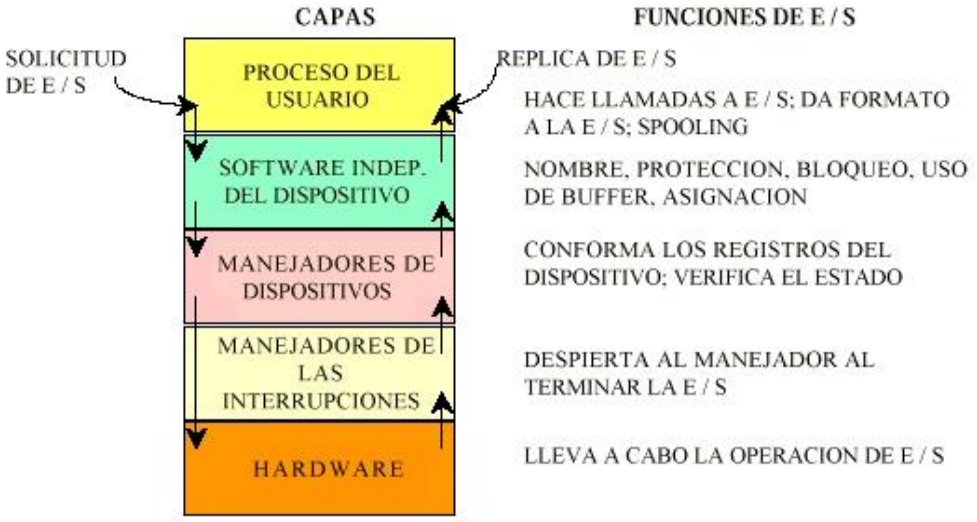


Figura 5.3: Capas del sistema de entrada / salida y las principales funciones de cada capa.

Manejadores de Interrupciones

Las interrupciones deben ocultarse en el S. O.:

- Cada proceso que inicie una operación de e / s se bloquea hasta que termina la e / s y ocurra la interrupción.
- El procedimiento de interrupción realiza lo necesario para desbloquear el proceso que lo inicio.

Manejadores de Dispositivos

Todo el código que depende de los dispositivos aparece en los manejadores de dispositivos.
Cada controlador posee uno o más registros de dispositivos:

- Se utilizan para darle los comandos.
- Los manejadores de dispositivos proveen estos comandos y verifican su ejecución adecuada.

La labor de un manejador de dispositivos es la de:

- Aceptar las solicitudes abstractas que le hace el software independiente del dispositivo.
- Verificar la ejecución de dichas solicitudes.

Si al recibir una solicitud el manejador está ocupado con otra solicitud, agregara la nueva solicitud a una cola de solicitudes pendientes.
La solicitud de e / s, por ej. para un disco, se debe traducir de términos abstractos a términos concretos:

- El manejador de disco debe:
    - Estimar el lugar donde se encuentra en realidad el bloque solicitado.
    - Verificar si el motor de la unidad funciona.
    - Verificar si el brazo está colocado en el cilindro adecuado, etc.
    - Resumiendo: debe decidir cuáles son las operaciones necesarias del controlador y su orden.
    - Envía los comandos al controlador al escribir en los registros de dispositivo del mismo.
    - Frecuentemente el manejador del dispositivo se bloquea hasta que el controlador realiza cierto trabajo; una interrupción lo libera de este bloqueo.
    - Al finalizar la operación debe verificar los errores.
    - Si todo está ok. transferirá los datos al software independiente del dispositivo.
    - Regresa información de estado sobre los errores a quien lo llamó.

- Inicia otra solicitud pendiente o queda en espera.

Software de E / S Independiente del Dispositivo

Funciones generalmente realizadas por el software independiente del dispositivo:

- Interfaz uniforme para los manejadores de dispositivos.
- Nombres de los dispositivos.
- Protección del dispositivo.
- Proporcionar un tamaño de bloque independiente del dispositivo.
- Uso de buffers.
- Asignación de espacio en los dispositivos por bloques.
- Asignación y liberación de los dispositivos de uso exclusivo.
- Informe de errores.

Las funciones básicas del software independiente del dispositivo son:

- Efectuar las funciones de e / s comunes a todos los dispositivos.
- Proporcionar una interfaz uniforme del software a nivel usuario.

El software independiente del dispositivo asocia los nombres simbólicos de los dispositivos con el nombre adecuado.

Un nombre de dispositivo determina de manera única el nodo-i de un archivo especial:

- Este nodo-i contiene el número principal del dispositivo, que se utiliza para localizar el manejador apropiado.
- El nodo-i contiene también el número secundario de dispositivo, que se transfiere como parámetro al manejador para determinar la unidad por leer o escribir.

El software independiente del dispositivo debe:

- Ocultar a los niveles superiores los diferentes tamaños de sector de los distintos discos.
- Proporcionar un tamaño uniforme de los bloques, por ej.: considerar varios sectores físicos como un solo bloque lógico.

Software de E / S en el Espacio del Usuario

La mayoría del software de e / s está dentro del S. O.

Una pequeña parte consta de bibliotecas ligadas entre sí con los programas del usuario.

La biblioteca estándar de e / s contiene varios procedimientos relacionados con e / s y todos se ejecutan como parte de los programas del usuario.

Otra categoría importante de software de e / s a nivel usuario es el sistema de spooling.

El spooling es una forma de trabajar con los dispositivos de e /s de uso exclusivo en un sistema de multiprogramación:

- El ejemplo típico lo constituye la impresora de líneas.
- Los procesos de usuario no abren el archivo correspondiente a la impresora.
- Se crea un proceso especial, llamado demonio en algunos sistemas.
- Se crea un directorio de spooling.

Para imprimir un archivo:

- Un proceso genera todo el archivo por imprimir y lo coloca en el directorio de spooling.
- El proceso especial, único con permiso para utilizar el archivo especial de la impresora, debe imprimir los archivos en el directorio.
- Se evita el posible problema de tener un proceso de usuario que mantenga un recurso tomado largo tiempo.

Un esquema similar también es aplicable para la transferencia de archivos entre equipos conectados:

- Un usuario coloca un archivo en un directorio de spooling de la red.

- Posteriormente, el proceso especial lo toma y transmite. Un ej. son los sistemas de correo electrónico.

Discos - Hardware Para Discos

Discos

Las siguientes son las principales ventajas con respecto del uso de la memoria principal como almacenamiento:

- Mucho mayor capacidad de espacio de almacenamiento.
- Menor precio por bit.
- La información no se pierde al apagar la computadora.

Un uso inapropiado de los discos puede generar ineficiencia, en especial en sistemas con multiprogramación.

Hardware Para Discos

Los discos están organizados en cilindros, pistas y sectores.
El número típico de sectores por pista varía entre 8 y 32 (o más).
Todos los sectores tienen igual número de bytes.
Los sectores cercanos a la orilla del disco serán mayores físicamente que los cercanos al anillo.
Un controlador puede realizar búsquedas en una o más unidades al mismo tiempo:

- Son las búsquedas traslapadas.
- Mientras el controlador y el software esperan el fin de una búsqueda en una unidad, el controlador puede iniciar una búsqueda en otra.

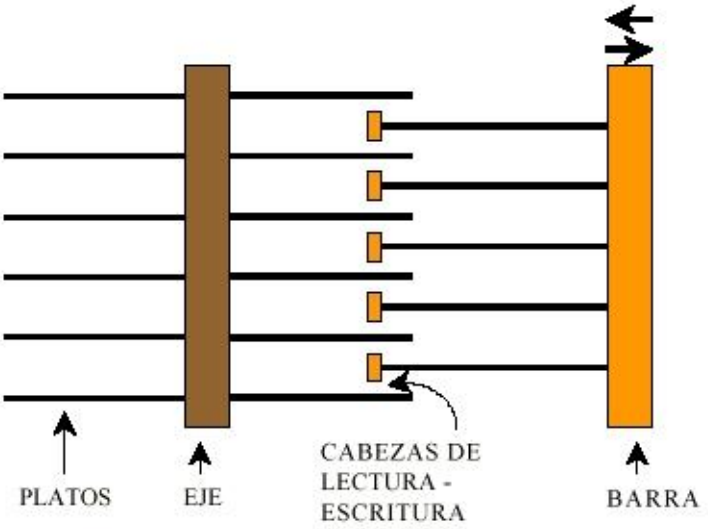
Muchos controladores pueden:

- Leer o escribir en una unidad.
- Buscar en otra.

Los controladores no pueden leer o escribir en dos unidades al mismo tiempo.
La capacidad de búsquedas traslapadas puede reducir considerablemente el tiempo promedio de acceso.

Operación de Almacenamiento de Disco de Cabeza Móvil

Los datos se graban en una serie de discos magnéticos o platos.
El eje común de los discos gira a una velocidad del orden de las 4.000 o más revoluciones por minuto.



Figura˜5.4: Esquema de un disco de cabeza móvil.

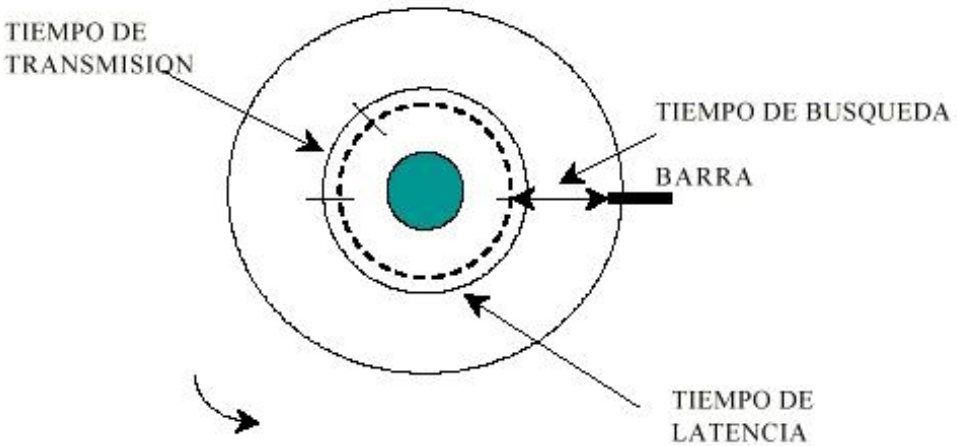
Se lee o escribe mediante una serie de cabezas de lectura - escritura (ver Figura 5.4):

- Se dispone de una por cada superficie de disco.
- Solo puede acceder a datos inmediatamente adyacentes a ella:
  - La parte de la superficie del disco de donde se leerá (o sobre la que se grabará) debe rotar hasta situarse inmediatamente debajo (o arriba) de la cabeza de lectura - escritura.
  - El tiempo de rotación desde la posición actual hasta la adyacente al cabezal se llama tiempo de latencia.

Todas las cabezas de lectura - escritura están montadas sobre una barra o conjunto de brazo móvil:

- Puede moverse hacia adentro o hacia afuera, en lo que se denomina operación de búsqueda.
- Para una posición dada, la serie de pistas accesibles forman un cilindro vertical.

A los tiempos de búsqueda y de latencia se debe agregar el tiempo de transmisión propiamente dicha (ver Figura 5.5).



Figura˜5.5: Componentes del acceso a un disco.

El tiempo total de acceso a un registro particular:

- Involucra movimientos mecánicos.

- Generalmente es del orden de centésimas de segundo, aunque el tiempo de latencia sea de algunas milésimas de segundo (7 a 12 aproximadamente).

https://www.youtube.com/watch?v=CBe0_QPS0u8

Algoritmos de Programación del Brazo del Disco

En la mayoría de los discos, el tiempo de búsqueda supera al de retraso rotacional y al de transferencia, debido a ello, la reducción del tiempo promedio de búsqueda puede mejorar en gran medida el rendimiento del sistema.
Si el manejador del disco utiliza el algoritmo primero en llegar primero en ser atendido (FCFS), poco se puede hacer para mejorar el tiempo de búsqueda.
Es posible que mientras el brazo realiza una búsqueda para una solicitud, otros procesos generen otras solicitudes.
Muchos manejadores tienen una tabla:

- El índice es el número de cilindro.
- Incluye las solicitudes pendientes para cada cilindro enlazadas entre sí en una lista ligada.
- Cuando concluye una búsqueda, el manejador del disco tiene la opción de elegir la siguiente solicitud a dar paso:
    - Se atiende primero la solicitud más cercana, para minimizar el tiempo de búsqueda.
    - Este algoritmo se denomina primero la búsqueda más corta (SSF: shor-test seek first).
    - Reduce a la mitad el número de movimientos del brazo en comparación con FCFS.

Ej. de SSF:

- Consideramos un disco de 40 cilindros.
- Se presenta una solicitud de lectura de un bloque en el cilindro 11.
- Durante la búsqueda, llegan solicitudes para los cilindros 1, 36, 16, 34, 9 y 12, en ese orden.
- La secuencia de búsqueda SSF será: 12, 9, 16, 1, 34, 36.
- Habrá un número de movimientos del brazo para un total de:
    - 111 cilindros según FCFS.
    - 61 cilindros según SSF.

El algoritmo SSF tiene el siguiente problema:

- El ingreso de nuevas solicitudes puede demorar la atención de las más antiguas.
- Con un disco muy cargado, el brazo tenderá a permanecer a la mitad del disco la mayoría del tiempo, como consecuencia de ello las solicitudes lejanas a la mitad del disco tendrán un mal servicio.
- Entran en conflicto los objetivos de:
    - Tiempo mínimo de respuesta.
    - Justicia en la atención.

La solución a este problema la brinda el algoritmo del elevador (por su analogía con el ascensor o elevador):

- Se mantiene el movimiento del brazo en la misma dirección, hasta que no tiene más solicitudes pendientes en esa dirección; entonces cambia de dirección.
- El software debe conservar el bit de dirección actual.

Ej. del algoritmo del elevador para el caso anterior, con el valor inicial arriba del bit de dirección:

- El orden de servicio a los cilindros es: 12, 16, 34, 36, 9 y 1.
- El número de movimientos del brazo corresponde a 60 cilindros.

El algoritmo del elevador:

- Ocasionalmente es mejor que el algoritmo SSF.

- Generalmente es peor que SSF.
- Dada cualquier colección de solicitudes, la cuota máxima del total de movimientos está fija, siendo el doble del número de cilindros.

Una variante consiste en rastrear siempre en la misma dirección:

- Luego de servir al cilindro con el número mayor:
  - El brazo pasa al cilindro de número menor con una solicitud pendiente.
  - Continúa su movimiento hacia arriba.

Algunos controladores de disco permiten que el software inspeccione el número del sector activo debajo del cabezal:

- Si dos o más solicitudes para el mismo cilindro están pendientes:
  - El manejador puede enviar una solicitud para el sector que pasará debajo del cabezal.
  - Se pueden hacer solicitudes consecutivas de distintas pistas de un mismo cilindro, sin generar un movimiento del brazo.

Cuando existan varias unidades, se debe tener una tabla de solicitudes pendientes para cada unidad.
Si una unidad está inactiva, deberá buscarse el cilindro siguiente necesario, si el controlador permite búsquedas traslapadas.
Cuando termina la transferencia actual se verifica si las unidades están en la posición del cilindro correcto:

- Si una o más unidades lo están, se puede iniciar la siguiente transferencia en una unidad ya posicionada.
- Si ninguno de los brazos está posicionado, el manejador:
  - Debe realizar una nueva búsqueda en la unidad que terminó la transferencia.
  - Debe esperar hasta la siguiente interrupción para ver cuál brazo se posiciona primero.

Generalmente, las mejoras tecnológicas de los discos:

- Acortan los tiempos de búsqueda (seek).
- No acortan los tiempos de demora rotacional (search).
- En algunos discos, el tiempo promedio de búsqueda ya es menor que el retraso rotacional.
- El factor dominante será el retraso rotacional, por lo tanto, los algoritmos que optimizan los tiempos de búsqueda (como el algoritmo del elevador) perderán importancia frente a los algoritmos que optimicen el retraso rotacional.

Una tecnología importante es la que permite el trabajo conjunto de varios discos.
Una configuración interesante es la de treinta y ocho (38) unidades ejecutándose en paralelo.
Cuando se realiza una operación de lectura:

- Ingresan a la cpu 38 bit a la vez, uno por cada unidad.
- Los 38 bits conforman una palabra de 32 bits junto con 6 bits para verificación.
- Los bits 1, 2, 4, 8, 16 y 32 se utilizan como bits de paridad.
- La palabra de 38 bits se puede codificar mediante el código Hamming, que es un código corrector de errores.
- Si una unidad sale de servicio:
  - Se pierde un bit de cada palabra.
  - El sistema puede continuar trabajando; se debe a que los códigos Hamming se pueden recuperar de un bit perdido.

Este diseño se conoce como RAID; siglas en inglés de “arreglo redundante de discos no costosos”.

Porqué es Necesaria la Planificación de Discos

En los sistemas de multiprogramación muchos procesos pueden estar generando peticiones de E/S sobre discos:

- La generación de peticiones puede ser *mucho más rápida* que la atención de las mismas:
  - Se construyen *líneas de espera o colas* para cada dispositivo.
  - Para *reducir el tiempo de búsqueda* de registros *se ordena la cola de peticiones:* esto se denomina **planificación de disco.**

La planificación de disco implica:

- Un examen cuidadoso de las peticiones pendientes para determinar la *forma más eficiente* de servirlas.
- Un análisis de las *relaciones posicionales* entre las peticiones en espera.
- Un *reordenamiento de la cola de peticiones* para servirlas *minimizando los movimientos mecánicos.*

Los tipos más comunes de planificación son:

- Optimización de la *búsqueda.*
- Optimización *rotacional (latencia).*

Generalmente los tiempos de búsqueda superan a los de latencia, aunque la diferencia disminuye:

- Muchos algoritmos de planificación se concentran en la *reducción de los tiempos de búsqueda* para un conjunto de peticiones.
- Generalmente la *reducción de la latencia* recién tiene efectos bajo *cargas de trabajo muy pesadas.*

Bajo condiciones de *carga ligera* (promedio bajo de longitud de la cola), es aceptable el desempeño del método *FCFS* (primero en llegar, primero en ser servido).
Bajo condiciones de *carga media o pesada,* es recomendable un *algoritmo de planificación de las colas de requerimientos.*

Características Deseables de las Políticas de Planificación de Discos

Los principales *criterios de categorización de las políticas de planificación* son:

- *Capacidad de ejecución.*
- *Media del tiempo de respuesta.*
- *Varianza de los tiempos de respuesta (**predecibilidad**).*

Una política de planificación debe intentar *maximizar la capacidad de ejecución:*

- Maximizar el número de peticiones servidas por unidad de tiempo.
- Minimizar la media del tiempo de respuesta.
- Mejorar el **rendimiento global**, quizás a costa de las peticiones individuales.

La planificación suele *mejorar la imagen total* al tiempo que reduce los niveles de servicio de ciertas peticiones:

- Se mide utilizando la *varianza de los tiempos de respuesta.*
- La varianza es un término estadístico que indica hasta qué punto tienden a desviarse del promedio de todos los elementos los elementos individuales.
- *A menor varianza mayor predecibilidad.*
- Se desea una política de planificación que minimice la varianza, es decir que *maximice la predecibilidad.*
- No debe haber peticiones que puedan experimentar niveles de servicio erráticos.

Optimización de la Búsqueda en Discos

Las estrategias más comunes de optimización de la búsqueda son las siguientes:

- FCFS.
- SSTF.
- SCAN.
- SCAN de N - Pasos.
- C - SCAN.
- Esquema Eschenbach.

Planificación FCFS (Primero en Llegar, Primero en Ser Servido)

Una petición no puede ser desplazada por la llegada de una petición con prioridad más alta.
No hay reordenamiento de la cola de peticiones pendientes.
Se ignoran las relaciones posicionales entre las peticiones pendientes.
Ofrece una varianza pequeña aunque perjudica a las peticiones situadas al final de la cola.

Planificación SSTF (Menor Tiempo de Búsqueda Primero)

El brazo del disco se sitúa en la siguiente petición que minimice el movimiento del brazo.
No respeta el orden de llegada de las peticiones a la cola.
Tiende a favorecer a las pistas del centro del disco.
La media de tiempos de respuesta tiende a ser más baja que con FCFS, para cargas moderadas.
Las varianzas tienden a ser mayores que con FCFS por el efecto de las pistas interiores y exteriores.

Planificación SCAN

El brazo del disco se desplaza sirviendo a todas las peticiones que encuentra a su paso.
Cambia de dirección cuando ya no hay peticiones pendientes en la dirección actual.
Ha sido la base de la mayoría de las estrategias de planificación implementadas.
Elimina las discriminaciones de SSTF y tiene menor varianza.
Las pistas exteriores son menos visitadas que las intermedias, pero no es tan grave como con SSTF.

Planificación SCAN de N - Pasos

La estrategia de movimiento del brazo es como en SCAN; solo da servicio a las peticiones que se encuentran en espera cuando comienza un recorrido particular.
Las peticiones que llegan durante un recorrido son agrupadas y ordenadas y serán atendidas durante el recorrido de regreso.
Posee menor varianza de los tiempos de respuesta si se compara con las planificaciones SSTF y SCAN convencionales.

Planificación C - SCAN (Búsqueda Circular)

El brazo se mueve del cilindro exterior al interior, sirviendo a las peticiones sobre una base de búsqueda más corta.
Finalizado el recorrido hacia el interior, salta a la petición más cercana al cilindro exterior y reanuda su desplazamiento hacia el interior.
No discrimina a los cilindros exterior e interior.
La varianza de los tiempos de respuesta es muy pequeña.

Esquema Eschenbach

El brazo del disco se mueve como en C - SCAN, pero:

- Las peticiones se reordenan para ser servidas dentro de un cilindro para tomar ventaja de la posición rotacional.
- Si dos peticiones trasladan posiciones de sectores dentro de un cilindro, solo se sirve una en el movimiento actual del brazo del disco.

Esta estrategia tiene en cuenta el retraso rotacional.

Conclusiones

Mediante trabajos de simulación y de laboratorio se demostró lo siguiente:

- La estrategia SCAN es la mejor con carga baja.
- La estrategia C - SCAN es la mejor con cargas medias y pesadas.
- La estrategia C - SCAN con optimización rotacional es la mejor para cargas muy pesadas (mejor que la estrategia Eschenbach inclusive).

Optimización Rotacional en Discos

En condiciones de carga pesada, las probabilidades de que ocurran referencias al mismo cilindro aumentan, por ello resulta útil considerar la optimización rotacional además de la optimización de búsqueda.
La optimización rotacional es de uso común en dispositivos de cabezas fijas.
La estrategia utilizada es la SLTF (tiempo de latencia más corto primero):

- Situado el brazo del disco en un cilindro:
  - Examina todas las peticiones sobre el cilindro.
  - Sirve primero a la que tiene el retraso rotacional más corto.

Consideraciones de los Discos Sobre los Sistemas

Los *principales interrogantes* son:

- Cuándo es útil la planificación de disco.
- Cuándo puede degradar el rendimiento.

El almacenamiento en disco como un recurso limitador
La planificación de disco puede *mejorar el rendimiento y eliminar el embotellamiento*, que se produce cuando se concentran grandes cargas de peticiones sobre relativamente pocos discos o pocos cilindros de un disco.
Nivel de multiprogramación
Generalmente la planificación es efectiva en *sistemas de tiempo compartido* con un nivel alto de multiprogramación.
Subsistemas de discos múltiples
Frecuentemente la cpu está conectada mediante canales (o bus) a dispositivos controladores, los que están conectados a las unidades de discos.
El *embotellamiento* puede producirse en algún disco, algún controlador o en algún canal.
Existe *software específico* para:

- Medir la actividad.
- Detectar dónde se produce el embotellamiento.

Para eliminar ciertos embotellamientos puede ser necesaria una *reconfiguración del hardware:*

- Agregar canales, controladores, dispositivos.
- Cambiar dispositivos de un controlador a otro.
- Cambiar controladores de un canal a otro.

Para ayudar a *reducir la congestión del canal*, muchos sistemas han incorporado la *técnica de examen (sensado) de posición rotacional (RPS):*

- *Reduce el tiempo* durante el cual un canal se encuentra ocupado en la búsqueda de un registro.
- *RPS permite al canal quedar libre* justo hasta antes de que el registro se encuentre debajo de la cabeza de lectura - grabación apropiada.
- *RPS permite varias peticiones activas* al mismo tiempo en un solo canal, incrementando la performance.

Distribución de peticiones no uniformes

Son muy comunes en *ciertas situaciones reales.*

Son frecuentes en procesos secuenciales de archivos secuenciales, para los que se afectaron cilindros adyacentes inmediatos.

Generalmente en estos casos las búsquedas son cortas y la planificación de disco será de poca utilidad.

Técnicas de organización de archivos

Los *métodos de organización y acceso de archivos*, así como los *DBMS* (manejadores de bases de datos):

- Son muy convenientes desde el punto de vista de las aplicaciones y del usuario.
- Pueden generar complicaciones en la implementación y el rendimiento, puesto que el recorrido de estructuras de índices, bloques de control, apuntadores, etc., puede significar un gran número de operaciones de e / s.

Manejo de Errores en Discos

Algunos de los *errores más comunes* en discos son:

- Error de programación: Ej.: Solicitar un sector no existente.
- Error temporal en la suma de verificación: Ej.: Provocado por polvo en la cabeza.
- Error permanente en la suma de verificación: Ej.: Un bloque del disco dañado físicamente.
- Error de búsqueda: Ej.: El brazo se envía al cilindro 6 pero va al 7.
- Error del controlador: Ej.: El controlador no acepta los comandos.

El manejador del disco debe controlar los errores de la mejor manera posible.

La mayoría de los *controladores:*

- Verifican los parámetros que se les proporcionan.
- Informan si no son válidos.

Respecto de los *errores temporales* en la *suma de verificación:*

- Generalmente se eliminan al repetir la operación.
- Si persisten, el bloque debe ser marcado como un **bloque defectuoso**, para que el software lo evite.

Otra posibilidad es que *controladores "inteligentes"* reserven cierta cantidad de pistas:

- Serán asignadas en reemplazo de pistas defectuosas.
- Una **tabla** asocia las *pistas defectuosas* con las *pistas de repuesto:*
    - Está alojada en la memoria interna del controlador y en el disco.
    - La sustitución es transparente para el manejador.
    - Puede afectarse el desempeño de los algoritmos de búsqueda, como el del elevador, ya que el controlador utiliza pistas físicamente distintas de las solicitadas.

Ocultamiento de Una Pista a la Vez en Discos

Generalmente el tiempo de búsqueda supera al de rotación y transferencia (aunque esto se está equilibrando).
Una vez resuelta la búsqueda del cilindro correspondiente, no es muy importante si se lee un sector o toda la pista:

- Especialmente en dispositivos con sensibilidad rotacional (RPS):
  - El manejador puede ver que sector se encuentra debajo de la cabeza y puede enviar una solicitud del siguiente sector:
    - Permite leer una pista en un tiempo de rotación.
    - De lo contrario se tardaría, en promedio, un tiempo de rotación más un tiempo de sector, para leer un solo sector.
- Algunos manejadores aprovechan esto mediante un caché secreto de una pista a la vez :
  - Es desconocido por el software independiente del dispositivo.
  - Si se necesita un sector del caché, no es necesaria una transferencia del disco.
  - Las principales desventajas de este ocultamiento de una pista a la vez son:
    - Complejidad del software.
    - Requerimientos de espacio para buffers.
    - Las transferencias del caché al programa que hace la llamada:
      - Las debe realizar la cpu mediante un ciclo programado.
      - No las puede hacer el hardware DMA.
  - Algunos controladores realizan el ocultamiento de una pista a la vez en su propia memoria interna:
    - Resulta transparente al manejador.
    - Las transferencias entre el controlador y la memoria pueden utilizar DMA.

Discos en RAM

Utilizan una parte de la memoria principal asignada con anterioridad para almacenar los bloques.
Tienen la ventaja del *acceso instantáneo:*

- No hay demora rotacional o debida a las búsquedas.
- Son adecuados para el almacenamiento de programas o datos con accesos muy frecuentes.

Los bloques de almacenamiento tienen el mismo tamaño que en los discos reales.
Cuando el manejador debe leer de o escribir en un bloque de un disco en RAM, calcula el lugar de la memoria donde se encuentra el bloque solicitado y lee o escribe en el mismo.

Relojes

Los *relojes* o *cronómetros* son esenciales para la operación de sistemas de tiempo compartido.
Registran la hora del día.
Evitan que un proceso monopolice la CPU.
El *software para reloj* toma generalmente la forma de un *manejador de dispositivo*, aunque *no es un dispositivo de bloque ni de caracter.*
Los relojes más sencillo trabajan con la línea de corriente eléctrica de 110 o 220 voltios y provocan una interrupción por cada ciclo de voltaje, a 50 o 60 hz.
Otro tipo de relojes consta de tres componentes:

- Un oscilador de cristal, un contador y un registro.
- Una pieza de cristal de cuarzo se monta en una estructura bajo tensión:

- Genera una señal periódica de muy alta precisión, generalmente entre 5 y 100 mhz.
- La señal se alimenta en el contador para que cuente en forma descendente hasta cero.
- Cuando el contador llega a cero, provoca una interrupción de la cpu.

Los *relojes programables* tienen varios modos de operación:

- *Modo de una instancia:*
  - Cuando el reloj se inicializa, copia el valor del registro en el contador.
  - Decrementa el contador en cada pulso del cristal.
  - Cuando el contador llega a cero provoca una interrupción y se detiene hasta ser nuevamente inicializado por el software.
- *Modo de onda cuadrada:*
  - Luego de llegar a cero y provocar la interrupción, el registro se copia de manera automática en el contador.
  - Todo el programa se repite en forma indefinida.
  - Las interrupciones periódicas se llaman *marcas del reloj.*

La ventaja del *reloj programable* es que su *frecuencia de interrupción puede ser controlada por el software.*

Las principales *funciones del software manejador del reloj* son:

- Mantener la hora del día o *tiempo real.*
- Evitar que los procesos se ejecuten durante más tiempo del permitido.
- Mantener un registro del uso de la cpu.
- Controlar llamadas al sistema tipo “alarm” por parte de los procesos del usuario.
- Proporcionar cronómetros guardianes de partes del propio sistema.
- Realizar resúmenes, monitoreo y recolección de estadísticas.

El software manejador del reloj puede tener que *simular varios relojes virtuales con un único reloj físico.*

Terminales

Las terminales tienen gran número de formas distintas:

- El *manejador de la terminal* debe ocultar estas diferencias.
- La parte *independiente del dispositivo* en el S. O. y los *programas del usuario* no se tienen que reescribir para cada tipo de terminal.

Desde el punto de vista del S. O. *se las puede clasificar en:*

- *Interfaz RS-232:*
  - Hardcopy (terminales de impresión).
  - TTY “de vidrio” (terminales de video).
  - Inteligente (computadoras con cpu y memoria).
- *Interfaz mapeada a memoria:*
  - Orientada a caracteres.
  - Orientada a bits.

Las *terminales RS-232* poseen un teclado y un monitor que se comunican mediante una *interfaz serial*, un bit a la vez; las conversiones de bits a bytes y viceversa las efectúan los chips uart (transmisores - receptores asíncronos universales).

Las *terminales mapeadas a memoria:*

- No se comunican mediante una línea serial.
- Poseen una interfaz mediante una memoria especial llamada *video RAM:*
  - Forma parte del espacio de direcciones de la computadora.
  - La cpu se dirige a ella como al resto de la memoria.
  - En la tarjeta de *video RAM* hay un chip llamado *controlador de video:*

- Extrae bytes del video RAM y genera la señal de video utilizada para manejar la pantalla.
- El monitor genera un rayo de electrones que recorre la pantalla pintando líneas.
- Cada línea está constituida por un cierto número de puntos o pixeles.
- La señal del controlador de video modula el rayo de electrones y determina si un pixel debe estar o no iluminado.
- Los monitores de color poseen tres rayos (rojo, verde y azul) que se modulan independientemente.

En las *pantallas mapeadas a caracteres:*

- Cada caracter en la pantalla equivale a dos caracteres de RAM:
  - Uno aloja al código (ASCII) del caracter por exhibir.
  - Otro es el byte de atributo, necesario para determinar el color, el video inverso, el parpadeo, etc.

En las *terminales mapeadas a bits:*

- Se utiliza el mismo principio.
- Cada bit en el video RAM controla en forma directa un solo pixel de la pantalla.
- Permite una completa flexibilidad en los tipos y tamaños de caracteres, varias ventanas y gráficos arbitrarios.

Con las pantallas mapeadas a memoria, el *teclado se desacopla totalmente de la pantalla:*

- El teclado dispone de su propio manejador.
- El manejador del teclado puede operar en modo caracter o en modo línea.

Las terminales pueden operar con una *estructura central de buffers* o con *buffers exclusivos para cada terminal.*

Frecuentemente *los manejadores de terminales soportan operaciones* tales como:

- Mover el cursor hacia arriba, abajo, a la izquierda o a la derecha una posición.
- Mover el cursor a x,y.
- Insertar un caracter o una línea en el cursor.
- Eliminar un caracter o una línea en el cursor.
- Recorrer la pantalla hacia arriba o hacia abajo “n” líneas.
- Limpiar la pantalla desde el cursor hacia el final de la línea o hasta el final de la pantalla.
- Trabajar en modo de video inverso, subrayado, parpadeo o normal.
- Crear, construir, mover o controlar las ventanas.

Administración de Dispositivos

La administración de dispositivos, es la administración de todos los recursos del hardware disponible, tanto los estándar que viene de fábricas, como las que se van agregando para hacer más poderosa o actualizar la PC. Todo dispositivo necesita presentarse al sistema operativo, agregando un pequeño programa que permite su uso. Este programa es llamado controlador. De aquí el controlador es un software que utiliza el sistema operativo para controlar el hardware.

La administración de dispositivos comprende 4 funciones básicas:

1. Controlar el estado de cada dispositivo (como unidades de cinta, unidades de disco, impresoras, graficadotes y terminales)
2. Utilizar políticas preestablecidas para determinar qué proceso obtendrá un dispositivo y durante cuánto tiempo.
3. Asignar los dispositivos. (A los procesos).
4. Desasignarlos en dos niveles: en el nivel de procesos cuando se ejecute un comando de entrada/salida (Temporal) y cuando el dispositivo se libera de manera permanente (Permanentemente).

Los dispositivos periféricos del sistema generalmente entran en una de tres clases:

- Dedicados
- Compartidos
- Virtuales

Dispositivos Dedicados: Se asignan sólo a un trabajo a la vez y le sirven todo el tiempo que está activo. La desventaja de los dispositivos dedicados es que se asignan a un usuario durante todo el tiempo que dure el trabajo que realiza, esto podría resultar ineficiente y es aún más ineficiente si el dispositivo no se utiliza el 100% del tiempo. Ejemplo: Impresoras, unidades de cinta.

Dispositivos Compartidos: Estos se puede asignar a más de un proceso a un mismo dispositivo. De forma que se puede compartir cualquier dispositivo de almacenamiento de acceso directo al entrelazar sus solicitudes, en estos casos el administrador de dispositivos tiene que controlar esta acción con bastante cuidado. Ejemplo: Discos Duros, DVD.

Dispositivos Virtuales: Son una combinación de los dispositivos dedicados y los compartidos; son dispositivos dedicados transformados en dispositivos compartidos. Además, estos son dispositivos que se pueden compartir por red, y utilizan Spooling, el cual genera una cola de espera en un buffer para el dispositivo. Ejemplo: Impresoras.

Los medios de almacenamiento se dividen en dos grupos:

- Medios de almacenamiento de acceso secuencial.
- Dispositivos de almacenamiento de acceso directo.

Medios de Almacenamiento de Acceso Secuencial.

El primer medio de almacenamiento fue el papel, pero su volumen y su precio hicieron que este medio resultara inaceptable para sistemas grandes. La cinta magnética se desarrolló para los primeros sistemas de cómputo de almacenamiento secundario rutinario.

Estos almacenan los registros en secuencia, uno después del otro. Este almacenamiento depende de la forma en que se decida almacenar los registros, esto es, en forma individual o en bloques. Si se almacena en forma individual después de cada registro le sigue un espacio que separa a cada registro almacenado, espacio entre registros (IRG, es aproximadamente de ½ pulgada de largo), sin importar el tamaño de los registros que separa. Esta forma es poco eficiente cuando los registros que se almacenan son de poco tamaño, ya que se pierde mucho en los IRG.

La otra forma de almacenamiento de registros es la de bloques, que consiste en agrupar los registros en bloques antes de registrarlos en la cinta. A esto se le llama “bloquear” y se lleva a cabo al crear el archivo. Los registros en un bloque son almacenados de forma secuencial, uno después del otro, y cada bloque es separado por un espacio que separa los diferentes bloques (IBG, al igual que el IRG es aproximadamente ½ pulgada de largo).

Medios de Almacenamiento de Acceso Directo (DASD).

Estos son los dispositivos que pueden leer o escribir en un lugar específico en un disco (también se conocen como dispositivos de almacenamiento de acceso aleatorio). Por lo general se agrupan en dos categorías principales:

a) Con cabezas de lectura y escritura fijas.
b) Con cabezas de lectura y escritura móviles.

DASD de Cabeza Fija.

Los primeros DASD eran tambores registrables magnéticamente. Estos eran en forma de una lata de café gigante, cubierta con película magnética y formato, de manera que las pistas corren a su alrededor. Los tambores de cabeza fija eran muy rápidos ya que utilizaban una cabeza de lectura/escritura para cada pista, pero también su valor era muy costoso y no contenían tantos datos como otros DASD, lo cual hizo que su popularidad descendiera. (Discos Magnéticos).

DASD de Cabeza Móvil.

Los tambores de cabeza móvil sólo tienen unas cuantas cabezas de lectura/escritura que se mueven de una pista a otra para cubrir la superficie del tambor. Mientras menos cabezas móviles tenga el tambor, menos es su valor. (Discos Ópticos).

Tiempo de acceso requerido: Según si un dispositivo DASD tiene cabezas fijas o móviles, tres factores pueden afectar el lapso requerido para tener acceso a un archivo:

- El tiempo de búsqueda
- El tiempo de latencia
- El tiempo de transferencia

Tiempo de búsqueda: es el más lento de los tres. Este es el intervalo requerido para colocar la cabeza de lectura/escritura sobre las pistas apropiadas. Este período no se aplica en dispositivos de cabeza fija.

Tiempo de latencia o retardo rotacional: Es el lapso necesario para girar el DASD hasta que el registro requerido pase por debajo de la cabeza de lectura/escritura.

Tiempo de transferencia: Es el más rápido de los tres, es cuando los datos realmente son transferidos del almacenamiento secundario a la memoria principal.

En Dispositivos de cabeza fija.

Estos dispositivos pueden tener acceso a un registro al conocer sus números de pista y de registro. El tiempo requerido para el acceso a los datos depende de dos factores:

1. La velocidad de rotación, aunque varía de un dispositivo a otro es constante en cada uno.
2. La posición del registro en relación con la posición de la cabeza de lectura/escritura.

El tiempo total de acceso es la suma del lapso de búsqueda más el periodo de transferencia.

Tiempo de búsqueda (retardo rotacional) + Tiempo de transferencia (transferencia de datos) = Tiempo de acceso

Ya que los DASD giran continuamente, hay tres posiciones básicas para el registro requerido relativo a la posición de la cabeza de lectura/escritura:

1. La mejor de la situación posible es cuando el registro se encuentra al lado de la cabeza de lectura/escritura cuando se ejecuta el comando de entrada/salida; esto da un retardo rotacional igual a cero.
2. La situación promedio es cuando el registro se encuentra en el otro extremo de la cabeza de lectura/escritura al momento que se ejecuta el comando de entrada/salida; esto da un retardo rotacional de t/2.
3. La peor de la situación es cuando el registro acaba de pasar por la cabeza de lectura/escritura cuando se ejecuta el comando de entrada/salida, con un retardo rotacional de t, ya que este necesita de una rotación completa para que el registro se coloque debajo de la cabeza.

En Dispositivos de cabeza móvil.

Estos dispositivos agregan el tercer elemento temporal al cálculo del periodo de acceso: el lapso requerido para mover el brazo a su posición bajo la pista adecuada o tiempo de búsqueda.

Tiempo de búsqueda (movimiento del brazo)

Tiempo de latencia (retardo rotacional) + Tiempo de transferencia (transferencia de datos) = Tiempo de acceso

El tiempo más largo entre los componentes del tiempo de acceso de esta ecuación es el de búsqueda. El tiempo de latencia y de transferencia es el mismo para los DASD de cabeza fija como para los de cabeza móvil.

Los dispositivos de cabeza móvil son más comunes que los de cabeza fija porque su costo es menor y tienen mayor capacidad, aun cuando el tiempo de recuperación es más largo.

Componentes del subsistema de entrada/salida.

Los componentes del subsistema de entrada/salida son: canales, unidad de control, dispositivos de entrada/salida.

Los canales de entrada/salida. Son unidades programables colocadas entre el CPU y las unidades de control. Estos controlan la velocidad rápida del CPU con la lenta del dispositivo entrada/salida y permite la superposición de operaciones de entrada/salida con las operaciones del procesador.

La unidad de control de entrada/salida es quien interpreta las señales que el canal envía para cada función. En la mayor parte de los sistemas una sola unidad de control está fija para varios dispositivos similares.

Al inicio de un comando de entrada/salida, la información que pasa del CPU al canal es:

1. Comando de entrada/salida (READ, WRITE, REWIND, etc..)
2. Número del canal
3. Dirección del registro físico que se va a transferir (desde el almacenamiento secundario o hacia él)
4. Dirección de inicio del buffer a partir del cual se va a transferir el registro o hacia el cual se va a mandar.

Comunicación entre dispositivos.

Para que un sistema de cómputo ocupado funcione eficientemente el administrador de dispositivos se apoya en varias características auxiliares y existen tres problemas por resolver:

1. Necesita saber qué componentes están ocupados y cuáles están libres
2. Debe ser capaz de aceptar las solicitudes que llegan durante el tráfico pesado de entrada/salida
3. Debe aceptar la disparidad de velocidades entre el CPU y los dispositivos de entrada/salida.

El primero se resuelve estructurando la interacción entre las unidades. Los dos últimos problemas se manejan colocando en memorias intermedias los registros y la cola de solicitudes.

Para saber cuando una operación ha terminado se usa una bandera de hardware que debe probar el CPU. Esta bandera está formada por 3 bits y esta se encuentra en el Channel Status Word (CSW). Cada bit representa uno de los componentes del subsistema de entrada/salida. Cada bit cambia de cero a uno para indicar que la unidad ha pasado de libre a ocupada y esta bandera puede ser accesada por cualquier componente para saber antes de seguir adelante con la siguiente operación de entrada/salida si la trayectoria está libre para su uso.

Dos maneras de realizar esta prueba son la encuesta y el uso de interrupciones.

- La encuesta utiliza una instrucción especial de máquina para probar la bandera. Esta manera tiene de desventaja que el CPU pierde tiempo en probar si un canal está libre o no y si tarda en efectuar este procedimiento podría que el dispositivo se deje de usar por un tiempo ya que el CPU cree que todavía está en uso.
- El uso de interrupciones es una forma más eficiente de probar las banderas. Ya que un mecanismo de hardware la prueba como parte de cada instrucción de máquina que ejecuta el CPU.

Acceso directo a la memoria (DMA). Es una técnica de entrada/salida que permite que una unidad de control tenga acceso directo a la memoria principal. Para esto el CPU envía suficiente información a la unidad de control y así se evita la intervención del CPU y puede continuar con otra tarea, mientras la unidad de control completa la transferencia.

Buffers. (también llamados memorias intermedias) son áreas temporales de almacenamiento que se encuentran en localidades convenientes en el sistema: memoria principal, canales y unidades de control. Estos se utilizan mucho para sincronizar mejor el movimiento de datos entre los dispositivos de entrada/salida relativamente lentos y un CPU muy rápido.

Administración de las solicitudes de entrada/salida.

El administrador de dispositivos divide la tarea en tres partes, cada una manejada por un componente específico de softwares del subsistema de entrada/salida:

- El controlador de tráfico de entrada/salida.
- El planificador de entrada/salida.
- El manejador de dispositivos de entrada/salida.

El controlador de tráfico de entrada/salida vigila el estado de los dispositivos, unidades de control y canales

Estrategias de búsqueda de manejo de dispositivos.

Una estrategia de búsqueda para el manejador de dispositivos de entrada/salida es la política predeterminada que utiliza para dar acceso al dispositivo a los diversos procesos que puedan estar esperándolo; define el orden en el cual los procesos obtienen el dispositivo, y la meta es mantener el tiempo de búsqueda en un mínimo.

Algunas estrategias de búsquedas más comunes:

- Primeras llegadas, primeros servicios (FCFS).
- Tiempo más breve de búsqueda primero (SSTF).
- SCAN y sus variaciones, LOOK, N-Step SCAN, C-SCAN, y C-LOOK.

Todo algoritmo de programación debe efectuar lo siguiente:

- Minimizar el movimiento del brazo.
- Llevar al mínimo el tiempo medio de respuesta.
- Minimizar la variación del tiempo de respuesta.

RAID

Es un juego de unidades físicas de disco considerado como una unidad lógica por el sistema operativo. Se introdujo para cerrar la brecha cada vez más grande entre procesadores más rápidos y unidades del disco más lentas.
RAID es la sigla para Redundant Array of Independent Disks. Su definición en español sería "Matriz Redundante de Discos Independientes". Se trata de una tecnología que combina varios discos rígidos (HD) para formar una única unidad lógica, donde los mismos datos son almacenados en todos los discos (redundancia). En otras palabras, es un conjunto de discos rígidos que funcionan como si fueran uno solo. Eso permite tener una tolerancia alta contra fallas, pues si un disco tiene problemas, los demás continúan funcionando, teniendo el usuario los datos a su disposición como si nada pasara. El RAID es una tecnología consolidada, que surgió de la Universidad de Berkesley, en California (EUA) a finales de la década de 1980.

Para conformar el RAID, es preciso utilizar por lo menos 2 discos rígidos. El sistema operativo, en este caso, mezclará los discos como una única unidad lógica. Cuando se graban datos, los mismos se reparten entre los discos del RAID (dependiendo del nivel). Con eso, además de garantizar la disponibilidad de los datos en caso de fallo de un disco, es posible también equilibrar el acceso a la información, de forma que no haya "cuellos de botella".

RAID nivel 0

Este nivel también es conocido como "Striping" o " Fraccionamiento". En él, los datos son divididos en pequeños segmentos y distribuidos entre los discos. Este nivel no ofrece tolerancia a fallos, pues no existe redundancia. Eso significa que un fallo en cualquiera de los discos rígidos puede ocasionar pérdida de información. Por esta razón, el RAID 0 es usado para mejorar la performance de la computadora, ya que la distribución de los datos entre los discos proporciona gran velocidad en la grabación y lectura de información. Mientras más discos existan, más velocidad es lograda. Esto porque, si los datos fueran grabados en un único disco, este proceso sería realizado en forma sequencial. Con El RAID, los datos que se guardan en cada disco son grabados al mismo tiempo. El RAID 0, por tener estas características, es muy usado en aplicaciones CAD y tratamiento de imágenes y vídeos.

RAID nivel 1

También conocido como "Mirroring" o " Espejado", el RAID 1 funciona añadiendo discos rígidos paralelos a los discos rígidos principales existentes en la computadora. Así, si por ejemplo, una computadora posee 2 discos, se puede anexar un disco rígido para cada uno, totalizando 4. Los discos que fueron añadidos, trabajan como una copia del primero. Así, si el disco principal recibe datos, el disco anexado también los recibe. De ahí el nombre de "espejado", pues un disco rígido pasa a ser una copia prácticamente idéntica del otro. De esa forma, si uno de los discos rígidos presenta una falla, el otro inmediatamente puede asumir la operación y continuar disponibilizando la información. La consecuencia en este caso, es que la grabación de datos es más lenta, pues es realizada dos veces. Sin embargo, la lectura de esa información es más rápida, pues puede ser accedida de dos fuentes. Por esta razón, una aplicación muy común del RAID 1 es su uso en servidores de archivos.

RAID nivel 2

Este tipo de RAID, adapta el mecanismo de detección de fallas en discos rígidos para funcionar en memoria. Así, todos los discos de la matriz están siendo "monitorizados" por el mecanismo. Actualmente, el RAID 2 es poco usado, ya que prácticamente todos los discos rígidos nuevos salen de fábrica con mecanismos de detección de fallas implantados.

RAID nivel 3

En este nivel, los datos son divididos entre los discos de la matriz, excepto uno, que almacena información de paridad. Así, todos los bytes de los datos tienen su paridad (aumento de 1 bit, que permite identificar errores) almacenada en un disco específico. A través de la verificación de esta información, es posible asegurar la integridad de los datos, en casos de recuperación Por eso y por permitir el uso de datos divididos entre varios discos, el RAID 3 logra ofrecer altas tasas de transferencia y confianza en la información. Para usar el RAID 3, por lo menos 3 discos son necesarios.

RAID nivel 4

Este tipo de RAID, básicamente, divide los datos entre los discos, siendo uno de esos discos exclusivo para paridad. La diferencia entre el nivel 4 y el nivel 3, es que en caso de falla de uno de los discos, los datos pueden ser reconstruidos en tiempo real a través de la utilización de la paridad calculada a partir de los otros discos, siendo que cada uno puede ser accedido de forma independiente. El RAID 4 es el indicado para el almacenamiento de archivos grandes, donde es necesario asegurar la integridad de la información. Eso porque, en este nivel, cada operación de grabación requiere un nuevo cálculo de paridad, dando mayor confianza al almacenamiento (a pesar de que esa operación torna las grabaciones de datos más lentas).

RAID nivel 5

Este es muy semejante al nivel 4, excepto por el hecho de que la paridad no está destinada a un único disco, sino a toda la matriz. Eso hace que la grabación de datos sea más rápida, pues no es necesario acceder a un disco de paridad en cada grabación. A pesar de eso, como la paridad es distribuida entre los discos, el nivel 5 tiene un poco menos de performance que el RAID 4. El RAID 5 es el nivel más utilizado y que ofrece resultados satisfactorios en aplicaciones no muy pesadas. Este nivel necesita de por lo menos 3 discos para funcionar.

RAID 0 + 1

El RAID 0 + 1 es una combinación de los niveles 0 (Striping) y 1 (Mirroring), donde los datos son divididos entre los discos para mejorar el ingreso, pero también utilizan otros discos para duplicar la información. Así, es posible utilizar el buen ingreso del nivel 0 con la redundancia del nivel 1. Sin embargo, es necesario por lo menos 4 discos para montar un RAID de este tipo. Estas características hacen del RAID 0 + 1 el más rápido y seguro, sin embargo es el más caro de ser implementado.

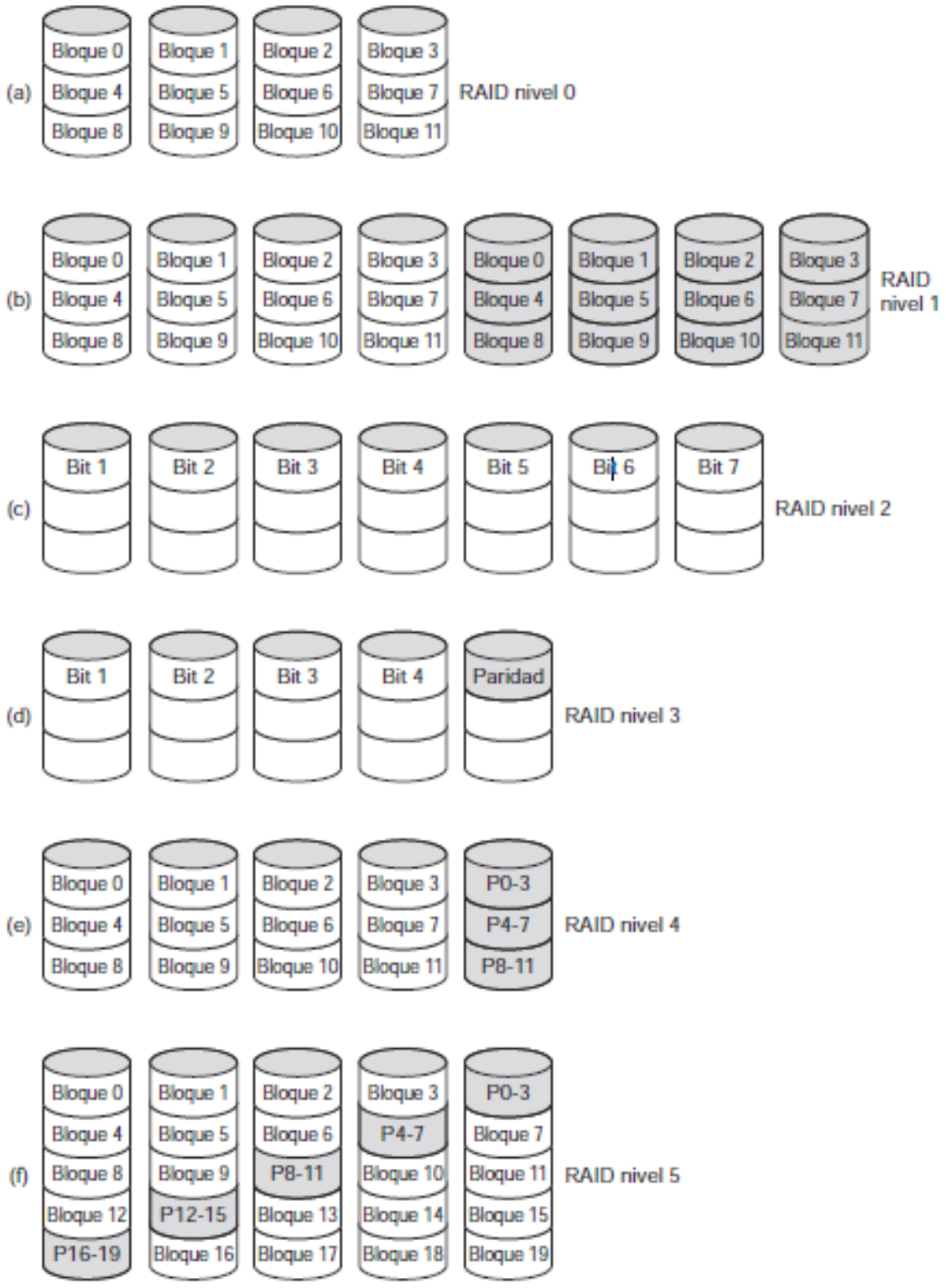


Figura 5-20. Niveles 0 a 5 de RAID. Las unidades de respaldo y de paridad se muestran sombreadas.

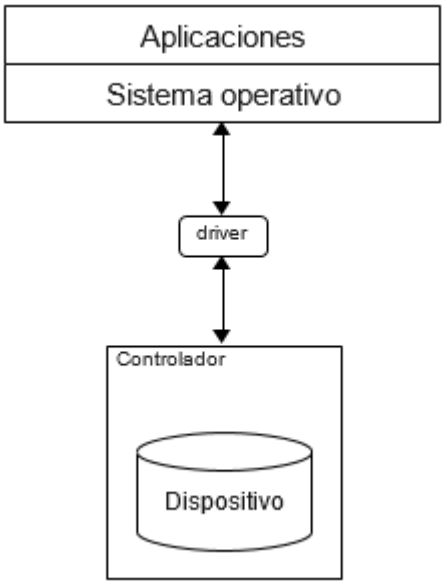
El sistema operativo: gestión de entrada/salida

Introducción y evolución histórica

El sistema de entrada/salida (E/S) es la parte del sistema operativo encargada de la gestión de los dispositivos de E/S. Así, actúa como interfaz entre los dispositivos y los usuarios, de manera los archivos y dispositivos se traten de una manera uniforme y puedan ser manipulados por medio de instrucciones de alto nivel.
La gestión de la E/S tiene asociados una serie de problemas:

- **Operación asíncrona:** Los eventos y sucesos asociados a la operación E/S no guardan ninguna relación de sincronismo con los de la CPU, por lo que la sincronización entre CPU y E/S no es trivial
- **Diferencia de velocidad:** La E/S es varios órdenes de magnitud más lenta que la CPU. Incluso hay grandes diferencias entre diferentes tipos de E/S (p. ej., es más rápida una red ethernet que una disquetera).
- **Diferencias de formato:** La CPU trabaja con representaciones de los datos y formatos fijas, mientras que cada dispositivo E/S lo gestiona de una manera diferente.
- **Heterogeneidad de los dispositivos:** Los dispositivos de E/S toman múltiples formas, lo que complica la gestión al tener que tener en cuenta todos los posibles tipos.

La conexión de los diferentes dispositivos de E/S al sistema, y la visión que tiene el sistema operativo y las aplicaciones de ellos, es la mostrada en la siguiente figura:



Se definen por tanto los siguientes conceptos básicos:

- **Dispositivo:** Es cada uno de los dispositivos de E/S que se van a gestionar.
- **Controlador:** Dispositivo hardware encargado de gestionar los aspectos internos de gestión del dispositivo (señales de control, etc.), ofreciendo al exterior un interfaz estandarizado. Este interfaz consiste en una serie de registros internos, que generalmente son:
    - **Registro de estado:** Indica el estado actual del dispositivo (ocupado, error, byte disponible, etc.)
    - **Registro de control:** Mediante escrituras al mismo, permite indicar la operación a realizar o cambiar aspectos de su configuración.
    - **Registros de entrada de datos:** Sirven para transferir datos hacia el dispositivo

    - **Registros de salida de datos:** Inverso, es decir, transferencia de datos desde el dispositivo hacia el SO.
- **Driver:** Módulo software encargado de intermediar entre el dispositivo físico y el sistema operativo, definiendo tres interfaces:
    - API para programas: Conjunto de operaciones que utilizarán los programas para acceder al dispositivo (lecturas, escrituras, etc.).
    - Comunicación con el controlador: Traducción de las operaciones de la API en operaciones del controlador, que serán enviadas para la ejecución de operaciones de E/S en el dispositivo.
    - Interfaz con el kernel: Comunicación entre el driver y el kernel, para su gestión interna. P. ej., para la carga de drivers, gestión de recursos, notificación de eventos, etc.

Esquemas hardware de gestión de E/S

Así, la principal tarea del controlador de E/S del sistema operativo será gestionar la comunicación entre los diferentes dispositivos. Para ello, existen distintos esquemas:

Polling

El controlador de E/S consulta de forma cíclica a todos los dispositivos, buscando si tienen operaciones pendientes. Es un esquema sencillo, si bien se pierde mucho tiempo de CPU consultando y esperando, especialmente en dispositivos lentos.
Admite dos enfoques:

- **Instrucciones específicas:** La CPU dispone de instrucciones especiales para el acceso a los dispositivos de E/S
- **E/S mapeada en memoria:** Se reserva un espacio del direccionamiento de memoria para los dispositivos de E/S, de manera que toda la información que éstos presentan en forma de registros, etc, se hace accesible leyendo esas posiciones de memoria. Igualmente, cualquier operación sobre el dispositivo se realiza escribiendo en las posiciones apropiadas de la memoria. El sistema operativo se encargará de traducir estas lecturas/escrituras a memoria en las operaciones reales sobre el dispositivo apropiado.

La ventaja de este método es que no requiere implementar instrucciones específicas en la CPU para la gestión de la E/S, ya que se utilizan las instrucciones convencionales de lectura/escritura en memoria..

Interrupciones

En este esquema, la CPU dispone de una línea extra llamada solicitud de interrupción (IRQ), que funciona de manera que, cuando se activa, la CPU detiene lo que esté haciendo y pasa a ejecutar una rutina especial, que actuará en función de los parámetros de la interrupción y que, una vez acabada, devolverá el control al flujo de programa que se estuviera utilizando.
Usando este esquema, cada dispositivo puede solicitar la atención de la CPU cuando lo necesite, de manera que no es necesario perder tiempo de la CPU monitorizando los dispositivos.
Para gestionar diferentes tipos de interrupción, en la solicitud de interrupción se indica un número correspondiente a diferentes tipos de interrupción. Así, el gestor de interrupción consultará una tabla o vector de interrupciones, y ejecutará la rutina indicada en la posición correspondiente.
Las interrupciones se clasifican generalmente en enmascarables o no enmascarables, refiriéndose a si la CPU puede inhibir la interrupción del código por una IRQ. Así, generalmente la CPU puede impedir, para zonas críticas del código, que una interrupción convencional interrumpa el flujo de ejecución. No obstante, las interrupciones no enmascarables se reservan para eventos de alta prioridad, para los que

no se acepta el enmascaramiento. Por ejemplo, una petición de un disco sería enmascarable, mientras que el aviso de un inminente corte eléctrico no lo sería.

Acceso directo a memoria (DMA)

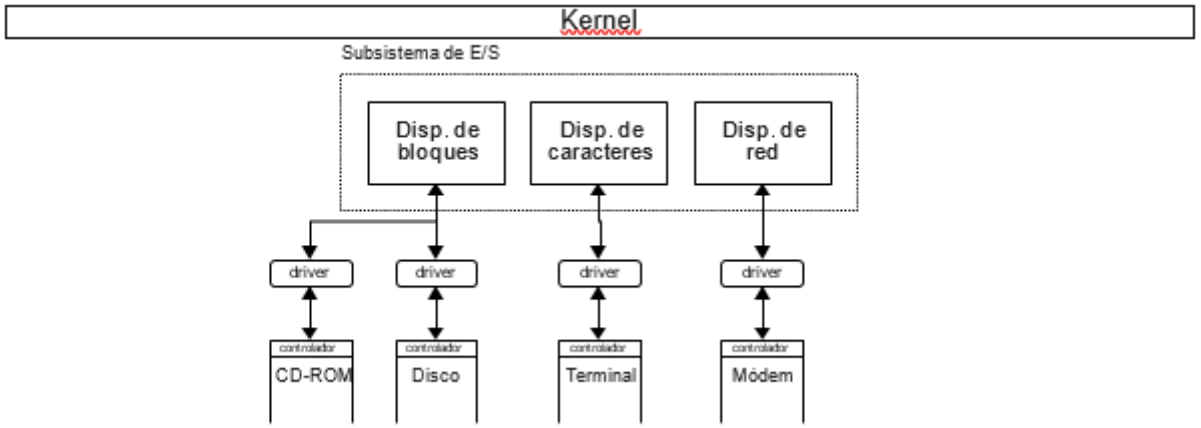
Para aquellos dispositivos que realizan grandes transferencias de datos, incluso el mecanismo de interrupciones es ineficiente, ya que hay que realizar una operación para cada dato. Así, un esquema común es, para estos casos, utilizar un procesador dedicado, llamado controlador DMA, que se encargará de los detalles de la transferencia.
La programación de este procesador consiste simplemente en indicarle el origen de los datos, la posición de destino en la memoria, y el número de datos a transferir, y a partir de aquí el procesador DMA se encargará de la comunicación con el dispositivo y con el bus de memoria. De esta forma, la CPU principal sólo intervendría al inicio de la operación, y al final, cuando el procesador DMA avisase a la CPU del fin de la transferencia mediante una interrupción.

Gestión de la E/S por parte del SO

A pesar de los numerosos tipos de dispositivos existentes y sus diferencias, el objetivo del tratamiento de la E/S del sistema operativo es tratar los dispositivos de una forma uniforme y estandarizada, de manera que las aplicaciones no deban preocuparse por los detalles de cada tipo de dispositivo.
Así, una estrategia común es catalogar los dispositivos en función de sus características principales:

- **Flujo de caracteres o de bloques:** Si transfieren los datos de carácter en carácter (p. ej. un módem) o en bloques de bytes (p. ej. un disco duro).
- De acceso aleatorio o secuencial
- **Síncrono o asíncrono:** Un dispositivo síncrono transfiere datos con tiempos de respuesta predecibles, mientras que uno asíncrono es irregular e impredecible a priori.
- **Compartido o dedicado:** Si puede ser usado por varios procesos a la vez, o debe ser usado en forma exclusiva.
- **Velocidad de operación:** En qué orden de magnitud realiza las transferencias.

Mediante esta catalogación, el subsistema de E/S del sistema operativo define una interfaz para el tratamiento de cada clase de dispositivos. Cada dispositivo necesitará únicamente de un pequeño programa o driver que implemente las operaciones de este interfaz. A continuación se muestra este esquema:



De esta forma se consiguen dos cosas:

- Abstraer los detalles de implementación de cada dispositivo, de modo que las aplicaciones no tienen por qué saber de qué dispositivo se trata.

- Simplificar la tarea de soporte de un nuevo dispositivo (es más sencillo hacer un driver que modificar el kernel).

Otro aspecto que clasifica a las operaciones de E/S es su sincronicidad:

- Operaciones bloqueantes (síncronas): Cuando se invocan, la ejecución de la aplicación se suspende hasta que finaliza la operación.
- Operaciones no bloqueantes (asíncronas): Una vez invocada, la aplicación continúa su tarea. La forma de funcionar se divide en dos tipos:
    - La operación devuelve el mejor resultado que tenga. P. ej., a una lectura de un socket se le puede especificar un timeout, y cuando expire devolverá los datos que tenga.
    - La operación se termina de ejecutar en 2º plano, y avisa a la aplicación principal cuando haya terminado y el subsistema de E/S requiera su atención.

Las operaciones asíncronas son más eficientes que las síncronas, ya que evitan las esperas activas (especialmente graves en dispositivos lentos), si bien el uso de operaciones asíncronas complica el código del programa.

Tipos de dispositivos

A continuación se muestran algunos detalles de las clases de dispositivos más comunes:

Dispositivos de bloques y caracteres

Los dispositivos de bloques y los de caracteres tienen un tratamiento muy similar, y se diferencian únicamente en la unidad de datos que manejan (bloque o carácter). Aparte de estas consideraciones, disponen de tres operaciones básicas:

- Leer
- Escribir
- Seek (sólo si permite acceso aleatorio)

Generalmente, un dispositivo de bloques ser utiliza de una de las siguientes formas:

- En bruto (raw): Se utilizan las tres operaciones básicas para leer/escribir bloques arbitrarios
- Como sistema de ficheros: Una capa superior del SO proporciona el acceso al dispositivo en forma de sistema de ficheros.
- Como mapeo en memoria: En lugar de leer/escribir explícitamente al dispositivo, se asigna una región de la memoria de manera que todas las lecturas/escrituras a esa zona se trasladan al dispositivo. De esta forma, desde el punto de vista del programa se puede usar el dispositivo de forma transparente, sin necesidad de usar ninguna operación de E/S.

Los dispositivos de caracteres se suelen utilizar mediante sus operaciones básicas, y se usan para aquellos dispositivos que generan/esperan flujos de datos lineales y relativamente asíncronos, como teclados, ratones o impresoras.

Dispositivos de red

Debido a su importancia, para los dispositivos de red, si bien podrían manejarse de forma genérica como un dispositivo de caracteres, se utiliza una abstracción específica, que es la de socket.
Un socket es un flujo de datos al que las aplicaciones pueden “conectarse”, realizando las siguientes operaciones:

- Conectarse a otro socket en otra máquina remota
- Leer o escribir paquetes de datos
- Recibir avisos de forma asíncrona cuando llegue información

Mediante el uso de sockets se facilita notablemente la gestión de las comunicaciones de red, con independencia del hardware de red o incluso de los protocolos subyacentes, maximizando la eficiencia y evitando situaciones de espera activa.

Mejoras del rendimiento

Buffering

Un buffer es un área de memoria en la que se almacenan los datos mientras se transfieren entre dos dispositivos o entre un dispositivo y una aplicación. Su utilidad es diversa:

- **Permiten hacer frente a las diferencias de velocidad entre dispositivos:** P. ej., la transferencia de un módem (lenta) puede acumularse en un buffer para ser escrita en un disco duro (rápido) de una sola vez, reduciendo así el número de operaciones necesarias.
- **Doble buffer:** En el caso de escrituras a un dispositivo, el mantener dos buffers de forma circular (uno con los datos que se escriben al dispositivo y otro con los que escribe la aplicación) permite mantener un flujo continuo de datos sin esperas.
- **Spooling:** Algunos dispositivos deben tratar con una unidad de datos completa, y no pueden hacerlo sólo con una parte, por lo que el uso de un buffer permite ir acumulando los datos hasta que se dispongan de suficientes para operar.
- **Lectura adelantada:** Cuando se realiza una lectura a un dispositivo lento, es común que el SO lea más datos de los necesarios y los coloque en un buffer. Así, si en el futuro se solicitan (cosa probable según los principios de localidad), ya se encuentran en memoria y se acelera la operación.
- **Caché:** Si una lectura solicita datos que se han leído anteriormente y se encuentran ya en un buffer, es posible leerlos directamente del buffer y ahorrarse una operación a E/S.

Políticas de planificación de discos

Debido a la importancia de los discos en los sistemas actuales como principal sistema de almacenamiento secundario, existen numerosas técnicas para acelerar su rendimiento.
Para comprender la motivación de muchas de estas técnicas, es necesario entender cómo funciona un disco duro. Básicamente, la información en un disco duro se almacena en pistas concéntricas de discos magnéticos, que a su vez se dividen en sectores. La información se lee y escribe mediante un cabezal. Así, una operación de disco se compone de tres etapas:

- **Seek:** Tiempo necesario para que el cabezal se posicione en la pista apropiada. Depende de la tecnología de fabricación del disco duro, y es, con diferencia, el mayor de los tres.
- **Latencia:** Una vez posicionado el cabezal, tiempo que transcurre hasta que el sector deseado pasa por el cabezal. Depende de la velocidad de rotación del disco.
- **Transferencia:** Tiempo que se tarda en recorrer todo el sector para leer/escribir su contenido. Depende de la velocidad de rotación y la densidad de los datos.

Generalmente, el tiempo de seek se mueve en torno a los 8-10 ms, mientras que el tiempo de latencia está en 3-4 ms. Una velocidad de transferencia típica podrían ser 20-30 MB/s, por lo que, por ejemplo, para leer 100 KB serían necesarios aproximadamente 3 ms.
Dado que el tiempo de posicionamiento del cabezal de un disco duro depende, además de las propias características físicas del disco, de la posición desde la que parta, es deseable ordenar convenientemente las operaciones de E/S para minimizar el movimiento del cabezal y mejorar, por tanto, el rendimiento del disco duro.

Existen numerosos esquemas de planificación de operaciones de disco:

- **FIFO:** Las peticiones se procesan en orden de llegada. No tiene un buen rendimiento, pero tampoco genera desigualdades entre las peticiones.
- **LIFO:** Siempre se procesa en primer lugar la última petición en llegar. Es mejor que FIFO, ya que aprovecha el principio de localidad que dice que la última petición es probable que sea a zonas del disco cercanas a las más recientes. Ahora bien, puede generar inanición si continúan llegando peticiones mientras otras quedan en espera.

Si desde el sistema operativo se tiene información sobre la posición actual del cabezal, es posible llevar a cabo políticas más complejas y eficientes:

- **SSTF (Shortest Seek Time First):** Encolar las peticiones, y enviar cada vez a disco la petición más cercana a la posición actual del cabezal. Es eficiente, pero puede provocar inanición al discriminarse peticiones a zonas lejanas.
- **SCAN:** Ordena las peticiones según accedan al principio o al fin del disco. Una vez finalizado un recorrido de principio a fin, hace otro desde el fin hasta el principio. El problema que tiene, a pesar de que no es posible la inanición, es que estadísticamente favorece a las peticiones de los extremos del disco. Una variación de SCAN es LOOK, en la que se cambia de dirección en cuanto ya no hay más operaciones en un sentido. Este algoritmo se conoce también como del ascensor, ya que emula su comportamiento.
- **SCAN circular:** Similar a SCAN, pero una vez acabada una pasada se vuelve a empezar otra vez desde el principio en lugar de hacer una pasada inversa. De esta forma el tiempo máximo de espera es globalmente menor y el uso de disco es equitativo.

Estas políticas pueden tener problemas ante la llegada de nuevos trabajos, que si se producen frecuentemente en las mismas zonas pueden ralentizar las pasadas y discriminar a otras peticiones. Para evitarlo, se pueden usar diferentes técnicas:

- SCAN de N pasos: En cada pasada se procesan únicamente N peticiones. Si durante una pasada llegan más, ponerlas en una cola y procesarlas después.
- FSCAN: Usar dos colas. En cada pasada se procesa una de ellas, encolando las nuevas peticiones en la otra. Al acabar la pasada, intercambiarlas y repetir el proceso.

Caché de disco

Como se ha comentado anteriormente, una técnica para aumentar el rendimiento es el uso de memorias caché para las operaciones de disco, de manera que se puedan evitar operaciones al disco si la información solicitada se encuentra en la caché.
La caché de disco puede implementarse de dos formas:

- Memoria específica en el propio dispositivo
- Reserva de parte de la memoria RAM como memoria caché

Generalmente se utiliza una combinación de ambos enfoques, dedicando una memoria de pequeño tamaño en el propio dispositivo, al tiempo que el SO reserva una cantidad variable de memoria principal como caché (en función de la carga del sistema, el tipo de dispositivo y otras consideraciones).
El uso de cachés de disco plantea dos problemáticas:

Sincronización con el disco

Si bien el uso de caché permite evitar operaciones al disco, también retrasa las escrituras, haciendo que el disco se encuentre en un estado inconsistente respecto a la memoria caché. Por tanto, una de las tareas del SO es asegurar que las escrituras en caché se trasladan en algún momento al disco duro, de forma que a largo plazo el disco duro quede en un estado consistente.

Para ello, hay dos enfoques:

- **Write-back:** Acumular escrituras a la memoria caché, trasladándolas al disco pasado un tiempo. Si bien esta técnica aumenta considerablemente la velocidad de las escrituras, también es peligrosa, ya que un fallo del sistema cuando aún no se han escrito los cambios dejaría al disco en un estado inconsistente.
- **Write-through:** Escribir los cambios a disco inmediatamente después de producirse. Es una opción más lenta al no permitir acumular varias operaciones, pero también es más segura al mantener el disco siempre consistente con la caché.

Políticas de reemplazo

Otro aspecto importante que debe gestionar el SO relacionado con las cachés de disco es qué bloques de la misma debe descartar cuando haya que cargar bloques nuevos. La incorrecta elección de los bloques a descartar puede provocar problemas importantes de rendimiento si, por ejemplo, un mismo bloque se descarta y se vuelve a cargar repetidamente.
Así, existen principalmente las siguientes políticas de reemplazo de bloques:

- **LRU (Least Recently Used):** Descartar el bloque que hace más tiempo que no se ha usado. Para ello, la caché se organiza como una pila, de manera que, cuando se accede a un bloque, se pone en la cabecera de la pila. Así, la implementación de LRU consiste simplemente en descartar el bloque que está en la parte inferior de la pila de bloques.
- **LFU (Least Frequently Used):** Descartar el bloque que ha sido usado con menos frecuencia. Para ello, junto con cada bloque se mantiene un contador que se incrementa con cada acceso. Así, al descartar se elige el bloque cuyo contador es más bajo.

Aunque en principio pueda parecer que LFU es una estrategia más apropiada, tiene problemas con las ráfagas de acceso a un bloque, ya que puede suceder que un bloque sea accedido muchas veces en un instante dado (incrementando así su contador), pero no vuelva a ser accedido posteriormente. El alto contador generado por la ráfaga de accesos impediría el descarte de este bloque.
Para solucionar este problema, se utiliza una técnica híbrida entre LRU y LFU, llamada descarte basado en frecuencia, que consiste en organizar los bloques en tres secciones:

Cuando un bloque es accedido, se incrementa su contador y se mueve a la zona alta. Si ya estaba en la zona alta, su contador no se incrementa, de forma que se evita que una ráfaga incremente los contadores artificialmente. Cuando transcurre un tiempo en la zona alta, el bloque se mueve a la zona media y, tiempo después, a la zona baja.
Para descartar un bloque, se escoge el que tenga el contador más bajo de la zona baja de la cola.

Gestión de dispositivos

En este capítulo se presenta un modelo general de la gestión de los dispositivos de entrada/salida basado en un esquema cliente-servidor. Como ejemplo de gestión de un dispositivo en particular se estudian los discos, en razón de que son los dispositivos soporte del sistema de ficheros, prestando atención preferente a la evaluación del rendimiento.

Introducción

Utilizaremos el término dispositivo para referirnos a cualquier elemento del computador que no sea el procesador o la memoria. Habitualmente los dispositivos se encargan de la entrada/salida, aunque la gestión de dispositivos hay que entenderla en sentido amplio, incluyendo los dispositivos de almacenamiento secundario y los de comunicaciones, e incluso la gestión del tiempo y de la energía. Esta heterogeneidad hace que el tratamiento de los dispositivos por el sistema operativo sea difícilmente generalizable para un estudio sistemático. En este capítulo vamos a intentar exponer un enfoque general de la entrada/salida, para después particularizar sobre un dispositivo concreto, centrándonos en los discos como dispositivo genérico para el soporte del sistema de ficheros.

Características de los dispositivos

Los dispositivos se caracterizan por su heterogeneidad, lo que introduce complejidad en el sistema operativo. Algunas de las características en las que los dispositivos pueden diferir son las siguientes:

- **Unidad de transferencia.** Unos dispositivos utilizan el byte como unidad de transferencia (dispositivos de caracteres, como el teclado o el ratón). Otros transfieren y/o almacenan la información en bloques (dispositivos de bloques, como discos y cintas magnéticas).
- **Velocidad.** Los rangos en los que se mueven los dispositivos son muy amplios. Los discos y los dispositivos de comunicación transfieren millones de caracteres por segundo y pueden hacerlo a velocidad constante, mientras que con el teclado se transfieren a lo sumo unos cuantos caracteres por segundo, con un periodo concreto impredecible.
- **Representación de los datos.** Incluso un mismo dispositivo puede utilizar diferentes codificaciones configurables en la instalación, como es el caso del teclado y el monitor.
- **Protocolos de comunicación.** La comunicación entre el dispositivo y la CPU se realiza de acuerdo a un determinado protocolo que depende del dispositivo y del bus de comunicación.
- **Operaciones.** Hay dispositivos de entrada, de salida y de entrada/salida. Además, algunos dispositivos requieren operaciones específicas (por ejemplo, posicionar el cabezal de lectura/escritura en los discos).
- **Errores.** Las condiciones de error varían con la naturaleza del dispositivo. Por ejemplo, en la impresora hay que tratar la falta de papel como una situación de error específica, mientras que en un disco puede haber errores en el posicionamiento del cabezal.

Para proporcionar una forma homogénea de direccionar los dispositivos, a nivel hardware éstos se conectan al sistema mediante controladores. El sistema operativo ya no trata con el dispositivo en sí mismo, sino con una interfaz que lo representa mediante un conjunto de direcciones o registros del controlador, que se pueden direccionar en el espacio de direcciones de memoria o constituir un espacio de direcciones independientes. El sistema se comunica con el controlador mediante operaciones de lectura/escritura sobre los registros de datos, estado y control, permitiendo tanto la transferencia de información como el diagnóstico y configuración del dispositivo. Estas operaciones las realizan las funciones de más bajo nivel del núcleo del sistema operativo, y son dependientes del hardware.

Tipos de entrada/salida

Por otra parte, y dependiendo en gran parte de las características del dispositivo, hay que distinguir tres tipos de entrada/salida, en función de cómo el sistema se sincroniza con el controlador:

- E/S programada. La sincronización es por encuesta, realizándose un bucle de espera activa en la consulta del registro de estado del controlador. Los sistemas operativos multiprogramados evitan este tipo de operación.
- E/S por interrupciones. El controlador activa una interrupción que permite la comunicación asíncrona del sistema operativo, que puede estar realizando otras tareas, con el dispositivo. Es la base que permite implementar un sistema operativo multiprogramado.
- E/S por DMA. Los dispositivos de bloques, que requieren una tasa de transferencia muy elevada, utilizan el acceso directo a memoria para las operaciones de entrada/salida, bien utilizando ciclos de memoria libres (robo de ciclo), bien adueñándose de los buses de memoria para transferir un bloque completo. Este tipo de entrada/salida implica la utilización de interrupciones para la sincronización con el fin de la transferencia.

La evolución del hardware ha llevado a incluir capacidad de proceso dentro del dispositivo (procesadores de E/S). El sistema operativo se comunica con el procesador de E/S para indicarle los parámetros de la operación a realizar y ordenar su inicio. El procesador de E/S ejecuta un código propio que controla los detalles de la operación. Por otra parte, lo habitual hoy en día es incluir una cierta cantidad de memoria RAM en el controlador o en el dispositivo, sobre la que el sistema operativo realiza la transferencia. Esto ocurre por ejemplo con los discos, las impresoras y las pantallas gráficas, que pueden contar con varios Mbytes de memoria.

Finalmente, un dispositivo puede estar accesible a través de una red, de forma transparente a las aplicaciones, situación habitual hoy en día, por ejemplo en las impresoras. Una pila de protocolos proporciona la comunicación entre la máquina cliente, que lanza la operación, y el servidor remoto, que gestiona el dispositivo. Este esquema cae fuera del ámbito de la asignatura objeto de estos apuntes.

Modelo general de la entrada/salida

Para facilitar su comprensión, el modelo de entrada/salida que vamos a presentar tiene como objetivo ser lo más sencillo y general posible, por encima de consideraciones acerca del rendimiento, e impone una estructura determinada al sistema operativo. Sus características fundamentales son:

- La entrada y salida se organiza y gestiona por capas, que responden a diferentes niveles de abstracción.
- El acceso a los recursos de entrada/salida se coordina de acuerdo al esquema cliente-servidor.

En el resto de esta sección estudiaremos ambas características como base del modelo general de entrada/salida. Los aspectos relacionados con la mejora del rendimiento pueden generalizarse en la utilización de buffers para el almacenamiento intermedio de la entrada/salida, que abordaremos en tercer lugar para completar el modelo.

Gestión de la entrada/salida por capas

Ante un panorama tan heterogéneo como el descrito en la introducción, abordar el estudio y diseño de la entrada/salida en un sistema operativo de manera comprensiva conduce a estructurar el sistema en diferentes niveles de abstracción o capas. Una capa L_k ofrece una interfaz a la capa superior, la capa L_{k+1}, conjunto de funciones que determinan la forma en que desde la capa L_{k+1} se accede a la capa L_k.

En un sistema operativo la capa de más arriba corresponde a los programas de usuario, y está ya fuera de sistema operativo. Como ya se conoce, desde esta capa se accede a la capa inferior, ya dentro del sistema operativo, mediante la interfaz de llamadas al sistema. La interfaz de una capa ha de estar bien definida para que el programador sepa a qué atenerse. Por ejemplo, en los sistemas UNIX la interfaz de llamadas al sistema está definida en la sección 2 del man.

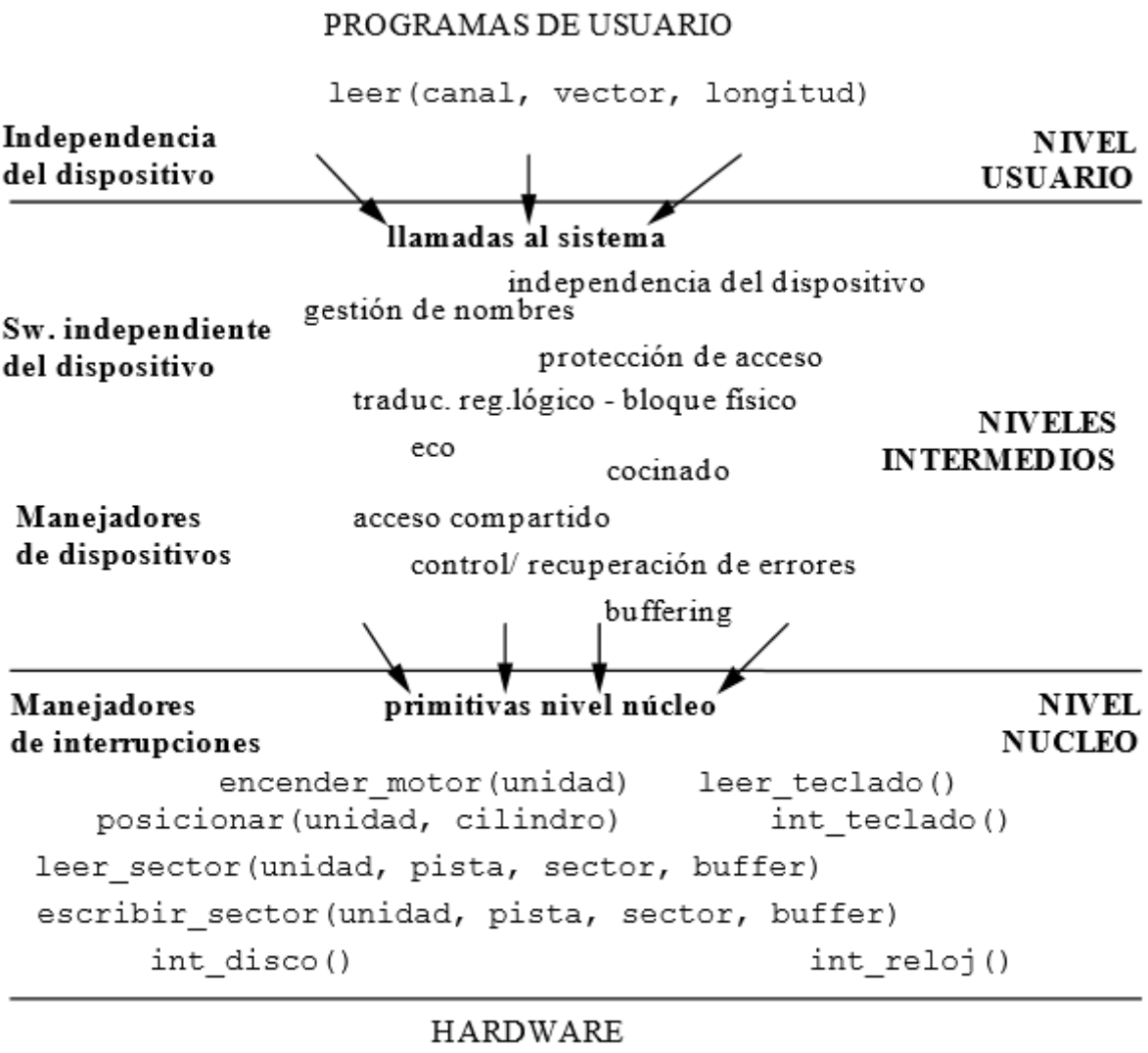


Figura 5.1. Gestión de dispositivos por capas

La estructura que vamos a proponer (Figura 5.1) consta de cuatro capas o niveles. De abajo arriba, el nivel más interno del sistema operativo (núcleo) programa los controladores de los dispositivos y maneja las interrupciones. Esta capa contiene software dependiente de los dispositivos y algunas partes han de ser codificadas en lenguaje máquina. Sobre el núcleo, en un segundo nivel se gestionan las peticiones de acceso a los dispositivos. Aquí residen los manejadores de los dispositivos (drivers), que tratan con las características particulares de los mismos y los controlan a través de las primitivas del núcleo. La tercera capa contiene software independiente del dispositivo: gestión de directorios, nombres, etc. Sobre estos niveles intermedios se monta la capa superior, que proporciona la interfaz de llamadas al sistema para las aplicaciones y muestra los dispositivos como abstracciones que se representan por canales, proporcionando conceptos como el redireccionamiento de la entrada-salida.

Esquema cliente-servidor

Las operaciones de entrada/salida se especifican desde las aplicaciones mediante las llamadas al sistema, que trabajan con canales o dispositivos lógicos. En general, una llamada al sistema típica (lectura o escritura) especifica de manera explícita o implícita los siguientes parámetros:

- La operación a realizar (leer, escribir…).
- El canal sobre el que se realiza la operación.
- La dirección (o posición) en el dispositivo E/S donde se accede. Normalmente está implícita (siguiente posición en un fichero) o incluso carece de sentido (lectura de teclado o ratón).
- La fuente o destino de la transferencia (dirección de memoria).
- La cantidad de información a transferir (longitud).
- En los sistemas que permiten operaciones síncronas y asíncronas, se indica esta condición y el evento con el que el programa que solicita la operación se va a sincronizar explícitamente.

El tratamiento de una operación de entrada/salida tiene dos partes. La primera, independiente del dispositivo, es el código utilizado por la llamada al sistema. Nos referiremos a ella como rutina de E/S. La segunda es el código del driver o manejador del dispositivo, y es dependiente del dispositivo. En nuestro modelo, la implementación del sistema operativo adopta el esquema cliente-servidor: las rutinas de E/S, ejecutadas por los procesos de usuario, corresponden a la parte del cliente del servicio, y el manejador, que se ejecuta como un proceso del sistema operativo, a la parte del gestor de la petición. A continuación presentaremos las estructuras de datos que proporcionan la interfaz entre las rutinas de E/S y los manejadores de los dispositivos, y que permiten hacer a la rutina de E/S independiente de las particularidades de la gestión del dispositivo.

Representación de la E/S

La estructura que proporciona la comunicación entre la rutina de E/S y el manejador del dispositivo se suele denominar IORB (Bloque de Petición de E/S, I/O Request Block). La rutina de E/S utiliza un IORB para cada petición. Contiene la siguiente información:

- Identificación del proceso cliente.
- Parámetros de la petición.
- Evento para la sincronización del cliente con el final de la operación.
- Diagnóstico de la operación, a establecer por el manejador de acuerdo al resultado de la operación.

En un sistema operativo donde toda la E/S fuera síncrona, cada proceso dispondría de un IORB único y privado, asociado a su PCB, y el evento puede ir implícito. En un modelo general donde también es posible la E/S asíncrona, cada proceso puede disponer de IORBs de un conjunto, que reservaría en exclusión mutua, y el evento de sincronización sería explícito. En este modelo general, los procesos se bloquean por eventos en vez de por operaciones de E/S.

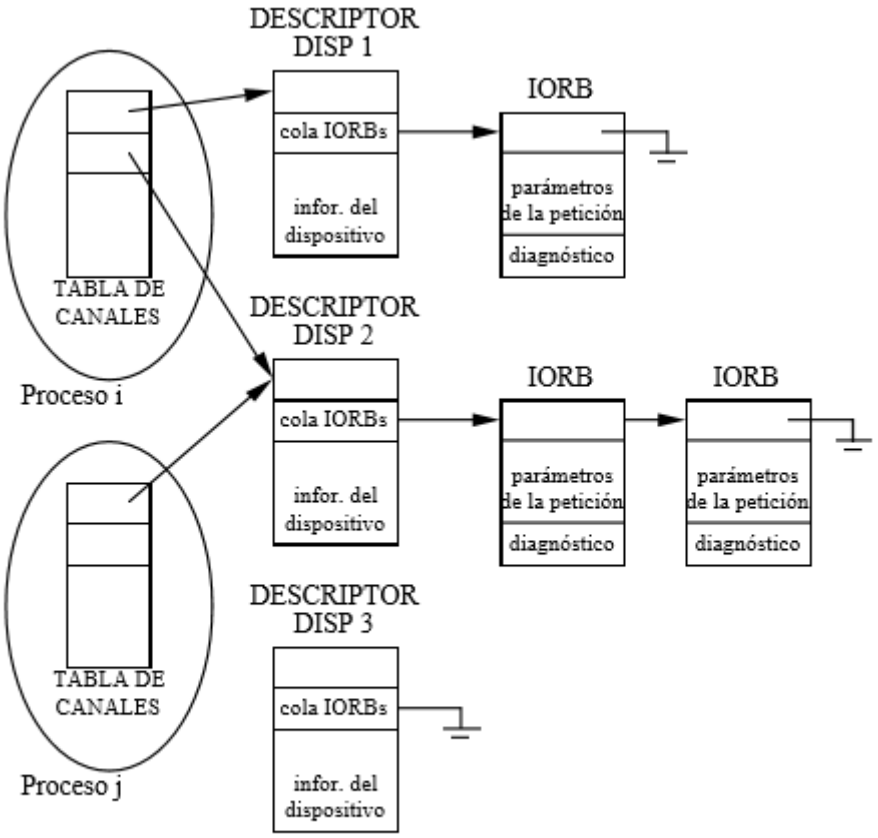


Figura 5.2. Un ejemplo del estado de la E/S en un sistema operativo

Las rutinas de E/S se mantienen independientes del dispositivo gracias a una estructura de datos asociada a cada dispositivo, el descriptor del dispositivo, que se direcciona a través de la tabla de canales y recoge las características del dispositivo y los parámetros propios de la operación con el dispositivo. Contiene información como:

- Estado del dispositivo.
- Modo de operación.
- Tablas de conversión.
- Apuntador a la cola de peticiones (IORBs) del dispositivo.
- Evento asociado al manejador correspondiente.

La Figura 5.2 representa un ejemplo del aspecto general de las estructuras de E/S en un sistema operativo de acuerdo al modelo introducido. Obsérvese cómo se integra en el modelo de colas de procesos descrito en el capítulo sobre la gestión de procesos.

Rutinas de E/S

Un proceso ejecuta una rutina de E/S poner una petición al manejador del dispositivo (driver). Gracias a que, como hemos visto, la tabla de canales del proceso especifica el descriptor del dispositivo correspondiente al driver, los detalles del manejador resultan transparentes a la petición, que tendrá el siguiente aspecto:

código_ret = petición_E/S (descriptor_del_dispositivo, operación, parámetros, evento_cliente)

El procedimiento que vamos a describir a continuación para ilustrar el funcionamiento de una rutina de E/S sigue un esquema que trata de ser general para todos los dispositivos (el acceso a ficheros requiere una etapa adicional de traducción de direcciones, como se verá en el capítulo correspondiente):

1. A partir de la entrada de la tabla de canales, tras comprobar las características de la operación y los permisos de acceso, la rutina de E/S accede al descriptor_del_dispositivo.
2. Construye la petición en un IORB libre con los parámetros de la petición.
3. Encola el IORB creado en la cola de peticiones asociada con el dispositivo (apuntada desde el descriptor del dispositivo).
4. Señala el evento de petición pendiente del manejador (especificado en el descriptor del dispositivo).
5. Si la operación es síncrona, espera al evento_cliente asociado a la petición (en sistemas síncronos es implícito al proceso y no se especifica en la petición).
6. Recoge el diagnóstico de la operación en el campo correspondiente del IORB. Interpreta las eventuales situaciones de error para devolver el código de retorno al proceso cliente.
7. Libera el IORB.

El Ejercicio 6 ilustra un ejemplo de código de rutina de E/S, con las definiciones asociadas.

Manejadores de dispositivos

Son los manejadores asociados a los dispositivos. Un manejador contiene código dependiente de las características del dispositivo, por lo que el esquema de funcionamiento que se proporciona aquí es muy general. Para cada petición, el manejador o gestor del dispositivo está a la espera de que la rutina de E/S señale su evento, según se ha descrito más arriba, para atenderla. En un esquema cliente/servidor, el manejador realiza un bucle infinito, bloqueándose en cada iteración. A continuación se describe el esquema ejemplo de una iteración para el tratamiento de una entrada/salida por interrupciones típica en un dispositivo de caracteres.

1. especificado en el IORB. Para algunos dispositivos es preciso traducir la representación de los Cuando el manejador detecta un evento de petición pendiente, toma un elemento (IORB) de su cola de peticiones y extrae los parámetros de la petición.
2. Programa la operación solicitada. Esto hace que el manejador se bloquee y que el núcleo del sistema operativo promueva un cambio de contexto.
3. Espera por el final de la operación. Cuando se ejecute la rutina de atención correspondiente, el manejador se desbloquea y puede continuar.
4. Transfiere la información a/desde el buffer datos.
5. Hace una comprobación de errores y escribe en el IORB el diagnóstico de la operación.
6. Señala el evento especificado por el cliente en el IORB.

Si se trata de transferencias DMA (en dispositivos de bloques), el paso (4) va implícito y no lo gestiona el manejador, realizándose entre los pasos (2) y (3). Más adelante estudiaremos el esquema de un manejador de discos como ejemplo de dispositivo de bloques.

Por otra parte, las características propias de cada dispositivo determinarán la naturaleza de alguno de los pasos, fundamentalmente en lo que se refiere al tratamiento de errores y la traducción de la representación de los datos.

En nuestro modelo, presente en sistemas operativos con estructura cliente-servidor o basada en micronúcleo, el manejador del dispositivo es un proceso que se desbloquea para atender las peticiones. Este esquema resulta conceptualmente sencillo, pero no es muy eficiente, por lo que los sistemas operativos tradicionalmente han buscado implementaciones más directas basadas en sincronización mediante eventos de dormir/despertar o semáforos. En los sistemas UNIX, el proceso que quiere realizar una operación de entrada/salida se bloquea en un evento asociado al dispositivo hasta que este se libera, pasando a modo sistema para ejecutar el código del manejador y restaurando el modo anterior

cuando finaliza la operación. De cualquier modo, el código del manejador ejecutado en este tipo de sistemas sigue los pasos descritos arriba para una iteración del servidor.

Almacenamiento intermedio de la E/S

Para desacoplar las velocidades de funcionamiento de los dispositivos con las de otros elementos del sistema y, por lo tanto, aumentar el rendimiento, es habitual la utilización de almacenamiento intermedio o buffering tanto de entrada como de salida.
El manejador del dispositivo proporciona buffers del sistema sobre los que se realiza la transferencia de entrada/salida. El buffer del sistema se copia a/desde el buffer de usuario, sobre el que la aplicación especifica la operación de E/S. El objetivo es doble. Por una parte, se amortigua la diferencia entre el ritmo de las peticiones de E/S y la capacidad del dispositivo para servirlas. Por otra parte, se consigue que, en sistemas con swapping y/o memoria virtual, las páginas de los programas de usuario que contienen los vectores especificados para la E/S no tengan que permanecer en memoria mientras el proceso está bloqueado[4]. Puede pensarse en diferentes esquemas de buffering, que dan lugar a diferentes formas de comportamiento de la entrada/salida en cuanto a la posibilidad de concurrencia entre las aplicaciones de usuario y los dispositivos (Figura 5.3):

A. **E/S sin buffer.** La transferencia se realiza directamente sobre el buffer de usuario, que debe quedar fijado en memoria. Si una aplicación pretende transferencias asíncronas, debe gestionarse sus propios buffers.
B. **Buffer simple.** No permite transferencias simultáneas, pero soporta transferencias asíncronas.
C. **Buffer doble.** Introduce concurrencia al permitir simultáneamente dos transferencias (usuario-sistema en un buffer y sistema-dispositivo en otro). Este esquema se puede generalizar a N buffers.
D. **Buffer circular:** permite soportar concurrencia del tipo productorconsumidor. Un ejemplo característico es el buffer de anticipación del teclado.

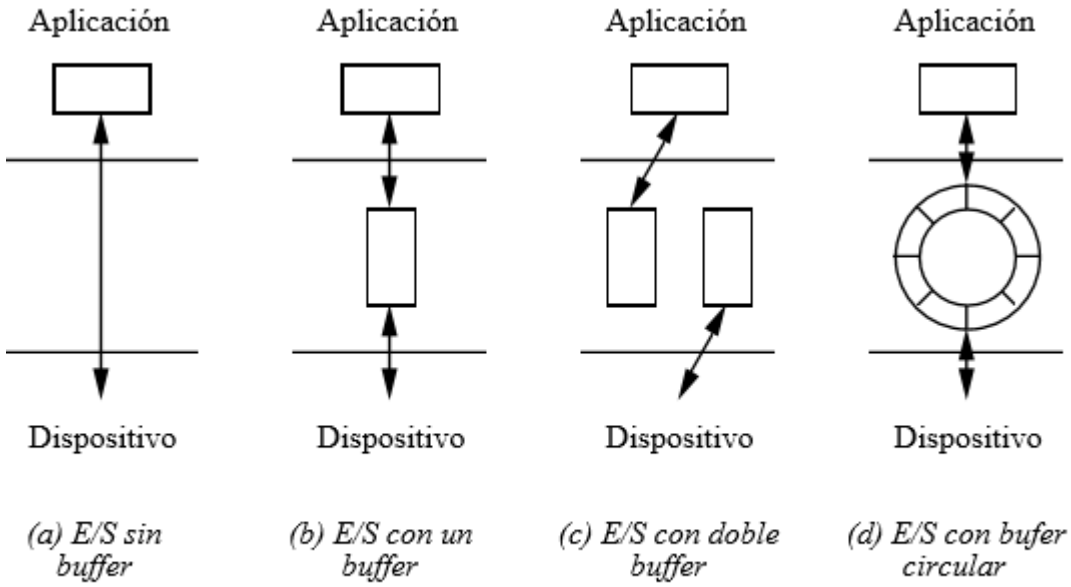


Figura 5.3. Esquemas de almacenamiento intermedio

La utilización de buffers mejora los parámetros de rendimiento temporal a cambio de espacio para implementarlos. También mejora la eficiencia de la CPU, aunque parcialmente compensada por el hecho de tener que copiar información entre los buffers del sistema y del usuario.

En algunos sistemas operativos la utilización o no de buffers se parametriza en la llamada al sistema de abrir y queda reflejada en la tabla de canales del proceso.

Gestión de discos

Estudiaremos en detalle la gestión del disco, dispositivo típico y común en los computadores de hoy en día, como ejemplo de las necesidades específicas que plantea al sistema operativo un dispositivo de cierta complejidad. Además los discos constituyen tradicionalmente el soporte de los sistemas de ficheros[5], por lo que la gestión de discos se relaciona con la gestión de ficheros, que se estudiará en el siguiente capítulo. Aquí nos centraremos en los aspectos más relevantes de la gestión del disco, que se implementan fundamentalmente en el manejador del dispositivo.

Hoy en día, los discos pueden ser de diferentes tecnologías, magnéticos (rígidos o flexibles) u ópticos (de sólo lectura, como CD y DVD, o también de escritura, como CD-R, CD-WR y los diferentes formatos de DVD grabables). En lo fundamental, la gestión es similar para todos ellos, por lo que la mayor parte de los conceptos que vamos a introducir se aplican a cualquier tipo de disco. Salvo que se especifique lo contrario, tomaremos como referencia los discos magnéticos rígidos, que son el soporte más habitual para el sistema de ficheros. En general un disco se organiza en sectores, que se agrupan en pistas. Los discos rígidos pueden tener más de dos superficies (se agrupan en un paquete de discos que giran con el mismo eje), y entonces se habla de cilindro como el conjunto de pistas superpuestas en la misma vertical. Los discos flexibles, en vías de desaparecer, pueden tener una o dos superficies o caras. Para generalizar, aquí hablaremos de la jerarquía sector-pistacilindro, aunque hay que advertir que lo habitual en los discos flexibles es utilizar la jerarquía equivalente sector-cara-pista; entendiéndose entonces por pista lo que antes hemos denominado cilindro.

Para que un disco sea utilizable en un sistema dado es necesario dar formato (o formatear), que incluye la inserción de información relativa al sistema de ficheros (como se verá en el siguiente capítulo), y para el control en los accesos a sectores.

El sector es la unidad de organización del disco visto éste como dispositivo físico. El sistema de ficheros agrupa los sectores en bloques, que es su unidad lógica de acceso y ubicación. Para la implementación del sistema de ficheros, el disco se ve como un vector de bloques[6].

El acceso a un sector del disco implica un conjunto de operaciones de posicionamiento:

(1) movimiento de los cabezales hasta alcanzar el cilindro al que se accede,
(2) selección del cabezal correspondiente a la pista, y
(3) espera de rotación hasta alcanzar el comienzo del sector.

Alguna de estas operaciones es de carácter mecánico, lo que introduce importantes retardos. Minimizar estos tiempos constituye un objetivo básico para la mejora el rendimiento. Una vez posicionado el sector correspondiente se lee/escribe de/en un buffer del controlador, donde se transfiere desde/hacia memoria principal. La inclusión de cierta cantidad de memoria RAM en los controladores modernos contribuye a minimizar el número de accesos al disco en sí, ya que los sectores accedidos pueden encontrarse en este buffer[7].

Dejaremos para la gestión de ficheros aspectos como la gestión de canales, la protección de accesos, la gestión de directorios y la ubicación de bloques (que se pueden resumir en la traducción de registro lógico de un fichero a número de bloque donde reside en disco), y que constituye la parte específica de la rutina de E/S en el tratamiento de ficheros. Aquí vamos a considerar estrictamente las funciones propias de la gestión del dispositivo disco:

- Arranque/parada automática del motor (en discos flexibles y ópticos).
- Traducción de número de bloque a cilindro, pista y sector.
- Planificación de accesos (gestión de peticiones).

- Control/recuperación de errores (con reintento).

Prestaremos especial atención a los aspectos relacionados con la evaluación del rendimiento y, finalmente, particularizaremos el esquema general cliente-servidor al manejo de los discos.

Arranque y parada del motor

Los discos rígidos giran continuamente y el motor sólo se para excepcionalmente (si se configura para ello), ya que los accesos suelen ser frecuentes. La mecánica de las unidades de discos rígidos es extremadamente precisa, no hay rozamientos[8] y disipan poco calor. En cambio, los discos flexibles y los ópticos no pueden estar girando continuamente. Como además los accesos a estos últimos suelen ser puntuales y menos frecuentes, lo normal es que el manejador encienda el motor cuando va a realizar un acceso. El apagado puede programarse tras un tiempo sin usar, lo que involucra a la rutina de atención al reloj.

Traducción de bloques

El manejador del disco recibe peticiones de acceso a bloques por parte de los procesos. Dado un número de bloque en el disco, b, éste se traduce a número de cilindro, c, número de pista en el cilindro, p, y número de sector en la pista, s. Si el sistema operativo maneja bloques de N sectores, la operación involucra el acceso a N sectores consecutivos, y los parámetros del disco obtenidos se refieren al primero de los sectores de ese bloque[9].

Dado un disco con una geometría de C cilindros de P pistas de S sectores (particularizado en el ejemplo de la Figura 5.4 con valores (C, P, S) = (3, 2, 4)) se deducen las fórmulas que permiten obtener los parámetros del acceso en función del número de bloque. El disco contará con C·P·S sectores y el número de bloques total, B, será un divisor entero de esta cantidad: B=CPS/N. Si "/" representa la división entera y "%" la operación módulo:

$$c = bN / PS$$

$$p = (bN / S) \% P$$

$$s = bN \% S$$

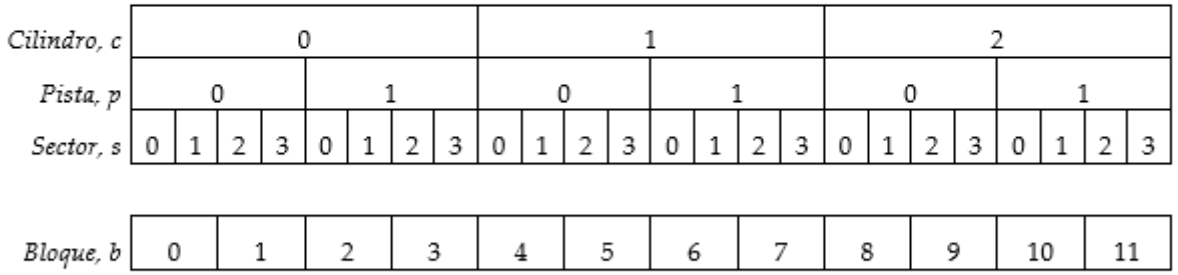


Figura 5.4. Representación de un disco con geometría (C, P, S) = (3, 2, 4). El tamaño de bloque, N, es de 2 sectores.

Reubicación de sectores. Interleaving de sectores.

En muchos casos, el valor de número de sector s obtenido de acuerdo a lo explicado anteriormente puede no corresponder al s-ésimo sector físico de la pista, sino referirse a un número de sector lógico que requiere aún una ulterior etapa en la traducción. Históricamente esto se justifica por lo siguiente. Un acceso al disco supone una transferencia DMA entre el dispositivo y la memoria. Supongamos una operación de lectura. Si el buffer del controlador del disco es de un tamaño tal que sólo puede contener un sector, el acceso al siguiente sector de la pista no podrá comenzar hasta que el buffer se haya transferido completo a memoria. Pero, durante la transferencia, el comienzo del siguiente sector habrá pasado debajo del cabezal de la unidad sin poder ser leído o escrito, y habrá que esperar a que se posicione en la siguiente rotación para accederlo.
Por este motivo, algunos controladores numeran los sectores de manera no correlativa en la pista, para que, accedido el sector i, el tiempo de posicionamiento en el i+1 se minimice. Esta técnica se conoce como interleaving de sectores. De acuerdo al tiempo de retardo producido por la transferencia de un sector, entre los sectores i e i+1 se intercala uno o más sectores, de forma que el tiempo de rotación para saltar estos bloques sea ligeramente superior al retardo de la transferencia.
Se denomina factor de interleaving al número de sectores que se intercalan entre dos sectores lógicamente consecutivos. Cabe hablar entonces de sectores lógicos y sectores físicos. La Figura 5.5(a) muestra la ordenación física de los sectores lógicos en una pista sin interleaving. La Figura 5.5(b) muestra cómo se ordenarían físicamente con un interleaving de factor 2. El factor de interleaving elegido será óptimo si minimiza el tiempo de posicionamiento en el siguiente sector.

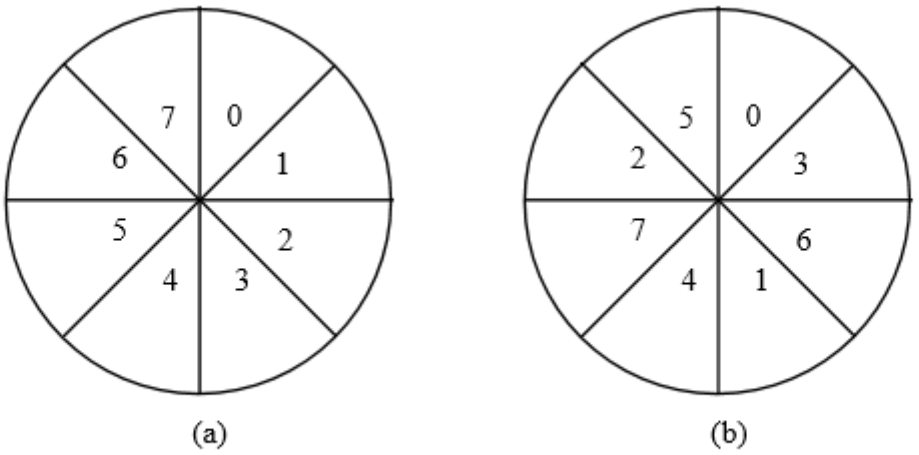


Figura 5.5. Ordenación de sectores lógicos: (a) sin interleaving, (b) con interleaving factor 2.

La forma de implementar el interleaving es mediante una tabla que proporciona la traducción de sectores lógicos a sectores físicos. Cuando es el controlador quien gestiona la traducción de sectores, se habla de interleaving hardware. Si el controlador no proporciona interleaving, el propio sistema operativo puede gestionarlo. En este caso se habla de interleaving software.
Hoy en día los controladores disponen de buffers de gran tamaño que permiten almacenar pistas enteras, por lo que la utilización del interleaving de sectores pierde su sentido. Sin embargo, la distinción entre sectores físicos y lógicos sigue teniendo interés, como veremos, para la gestión de sectores dañados.

Planificación de accesos

El acceso a un bloque en disco consta de varias operaciones con retardos relacionados con las características físicas del dispositivo. Mientras que la selección de cabezal es una operación de carácter eléctrico y, por lo tanto, presenta un retardo nulo, el posicionamiento en un cilindro y la espera de rotación son operaciones de carácter mecánico cuyo retardo es importante y debe minimizarse para mejorar el rendimiento del sistema. Este es uno de los objetivos del manejador del disco. Globalmente, la estrategia afecta al tiempo de rotación y de movimiento del cabezal.
La espera de rotación se reduce con el interleaving de sectores o utilizando buffers del tamaño adecuado. El posicionamiento en cilindro es, en un sistema multiprogramado, un problema de ordenación de las peticiones.
Ordenando las peticiones se puede minimizar el tiempo medio de posicionamiento del cabezal en el comienzo de un bloque. Aunque el orden afecta también al tiempo de rotación, aquí vamos a considerar únicamente estrategias basadas en el número de cilindro. Combinando ambas operaciones pueden concebirse políticas globales más complejas.
Supondremos que el manejador de disco, en un instante de tiempo dado, dispone de una cola de peticiones de acceso a disco. Cada petición se especifica como el número de cilindro al que se quiere acceder. Se trata de reordenar las peticiones de la cola de forma que la distancia recorrida por el cabezal sea la menor. Aunque el tiempo de traslación del cabezal no es lineal con la distancia, consideraremos esta aproximación suficiente. Tenemos las siguientes políticas de gestión:

- FCFS o FIFO: No requiere ningún tipo de proceso sobre la cola de peticiones. Es la más sencilla pero poco eficiente.
- SSF (el más cercano primero): No garantiza que el cabezal recorra la distancia mínima. Su principal problema deriva del hecho de que los cilindros centrales son los más cercanos en media a los demás cilindros. Como la cola de peticiones es dinámica, esto puede producir inanición en los cilindros extremos.
- SCAN o LOOK (algoritmos del ascensor): El cabezal recorre los cilindros alternativamente en sentido ascendente y descendente, atendiendo todas las peticiones por las que pasa[10]. Beneficia a los cilindros centrales, por lo que esta estrategia se combina con la de ubicar allí los bloques de acceso más frecuente.
- C-SCAN o C-LOOK (algoritmos del ascensor de sentido único): Es una modificación del algoritmo del ascensor para evitar la discriminación de los cilindros. Ahora el cabezal recorre los cilindros en un único sentido (ascendente), volviendo al cilindro 0 (operación de recalibrar) entre cada recorrido.

Tratamiento de errores

Por su naturaleza mecánica, los discos están expuestos a numerosas causas de error. Es importante que las primitivas del núcleo del sistema operativo devuelvan un código que permita tratar el error para, si es posible, recuperar la operación.
Los errores más comunes en los accesos al disco se pueden englobar en alguno de los dos tipos siguientes:

- Errores de posicionamiento: el cabezal no se sitúa sobre el cilindro solicitado. Es necesario recalibrar, lo que supone poner el cabezal en el punto de referencia (cilindro 0), para poder reintentar el posicionamiento.
- Errores de control de paridad, que pueden indicar la existencia de un sector dañado o la presencia de suciedad en el cabezal o la superficie del disco.

En cualquier caso el gestor del disco tiene la opción de reintentar la operación un número razonable de veces. Si finalmente se consigue realizar la operación, el error se considera transitorio y el usuario no llega a tener noticias del percance (podría tratarse de una simple partícula de polvo que terminó

desapareciendo). Si el problema persiste (p. ej. si la causa fuera un sector dañado), el error se considera permanente y se transmite a los niveles superiores del sistema operativo. Los discos de hoy en día introducen mecanismos automáticos para tratar los errores permanentes, como la reubicación de sectores dañados. Una tabla hace corresponder cada sector lógico dañado a un sector físico en otra ubicación del disco. Obsérvese que esta técnica es análoga a la de interleaving, con la diferencia de que la correspondencia entre sectores lógicos y físicos es global a todo el disco.

Evaluación del rendimiento

Como se ha visto más arriba, el acceso y transferencia de un determinado bloque del disco conlleva una sucesión de operaciones con retardos asociados, algunos de ellos considerables al ser de naturaleza mecánica. Los tiempos que componen una operación de acceso a un bloque son los siguientes:

1. Tiempo de posicionamiento sobre el cilindro, tpos. Depende del recorrido que tenga que hacer el brazo de cabezales (distancia al cilindro desde la posición actual). Para la estimación del tiempo de acceso vamos a suponer un tiempo medio (que denotaremos Tpos), dependiente de la estrategia de planificación de accesos que se siga.
2. Selección de pista. Se trata de seleccionar uno de los cabezales del brazo. Como únicamente supone la activación de una señal eléctrica, este retardo no se considera.
3. Tiempo de posicionamiento en el comienzo del primer sector del bloque. Está relacionado con la velocidad de giro y varía desde 0 al tiempo de un giro completo, tgiro. En media hay que considerar este retardo como de 0,5tgiro.
4. Tiempo de posicionamiento en el resto de los sectores (para bloques de más de un sector). Los accesos a los demás sectores están condicionados por la posición del cabezal tras la transferencia del sector anterior, lo que depende del factor de interleaving o, alternativamente, de que el tamaño del buffer permita almacenar todos los sectores del bloque. En un disco de S sectores, sin buffers y con factor de interleaving F, el tiempo de posicionamiento en cada uno de los N–1 últimos sectores de un bloque de N sectores será Ftgiro/S.
5. Tiempo de acceso a cada sector. Depende de la velocidad de giro y del número de sectores por pista. Para cada sector del bloque es tgiro/S.
6. Tiempo de transferencia de sector, ttrans. Involucra una operación de DMA y depende de múltiples factores, como la velocidad del bus, la existencia o no de buffers en el controlador o el funcionamiento del DMA. Este tiempo se solapa con el posicionamiento en el siguiente sector, de forma que sólo hay que considerar el tiempo de transferencia de un sector por bloque, independientemente de su tamaño.

Además, habría que considerar la incidencia de los retardos debidos al tratamiento de errores (por ejemplo, puede ser necesario posicionar más de una vez el cabezal), que depende de las probabilidades de cada tipo de error. Simplificando, para bloques de N sectores, con factor de interleaving F, sin considerar el tratamiento de errores, ni la existencia de buffers en el controlador, ni solapamiento entre operaciones, se obtiene la siguiente expresión del tiempo medio de acceso al bloque:

$$Tacc = Tpos + (0,5 + (N–1)F/S + N/S)\ tgiro + ttrans$$

Hay que insistir en que esta expresión responde a un modelo simplificado con las condiciones comentadas. En otros casos, la expresión del tiempo de acceso debe deducirse de acuerdo a los parámetros definidos para el sistema particular.

Esquema de un manejador de discos

Habitualmente, los programas de usuario acceden al disco indirectamente a través del sistema de ficheros. Las rutinas de entrada/salida resuelven la independencia del dispositivo, traducen los parámetros de la llamada al sistema a un número de bloque absoluto y, en su caso, ordenan al manejador del dispositivo el acceso al bloque. Los detalles de este proceso se verán en el capítulo de gestión de ficheros. El manejador del disco es el responsable de gestionar el acceso a los sectores correspondientes del disco. Resumiremos aquí el esquema de funcionamiento de un manejador de discos, particularizando el esquema general introducido en la Sección anterior.

1. Cuando el manejador detecta un evento de petición pendiente, toma un elemento (IORB) de su cola de peticiones y extrae los parámetros de la petición. Se traduce el número de bloque absoluto a los parámetros del disco: cilindro, pista y sectores. Si hubiera más de una petición pendiente, el manejador aplica una de las políticas de planificación de accesos estudiadas en la Sección 5.3.4.
2. Si el motor estuviese parado, el manejador programa el encendido de la unidad y se bloquea por tiempo hasta que el disco ha alcanzado el régimen de revoluciones permanente[11].
3. Si el cabezal no estuviese colocado sobre el cilindro a acceder, se programa una operación de posicionar. El manejador queda bloqueado hasta la finalización de la operación. Como esta maniobra está sujeta a errores, es habitual, en su caso, reintentarla después de una operación de recalibrado.
4. Para cada uno de los sectores del bloque, programa la transferencia DMA correspondiente, en uno u otro sentido dependiendo de la operación (lectura o escritura). Cada transferencia implica el bloqueo del manejador y la consiguiente comprobación de errores tras la finalización de la transferencia (señalada por la rutina de atención a la interrupción de DMA), con la posibilidad de reintento.
5. El manejador escribe en el IORB el resultado de la operación.
6. El manejador señala el evento especificado por la rutina de E/S en el IORB.

Sistemas de Entrada/Salida

Las aplicaciones utilizan los dispositivos (devices) para realizar la I/O (entrada-salida). Estos dispositivos son variados y trabajan de manera diferente: secuencialmente, random; transfieren datos asincrónicamente o sincrónicamente; pueden ser de sólo lectura (read-only) o lectura-escritura (read-write), etc.
El sistema operativo debe permitir que las aplicaciones puedan utilizar esos dispositivos, proveyendo una interfaz que los presente de la manera más simple posible.
Los dispositivos son una de las partes más lentas de un sistema de computo. Por lo tanto, el SO, debe manejar la situación como para salvar esa diferencia de velocidad.
La función de un SO en los sistemas de I/O, es manejar y controlar las operaciones y los dispositivos de I/O.

La aplicación y la I/O

EL SO debe ofrecer al resto del sistema una interface standard, simple y uniforme para el uso de un dispositivo.
La aplicación trata de abrir un archivo de un disco, abstrayéndose del tipo de disco que es. Una interface define un conjunto de funciones estandarizadas que permite la abstracción, el encapsulamiento y la división del software en capas.
Los device drivers son módulos del kernel que si bien internamente diferencian entre los distintos tipos de dispositivo, ofrecen al sistema interfaces estándar.
Veamos la estructura en capas de software de la parte del kernel relacionada con la I/O.

<table>
<tr><td colspan="7">Kernel</td></tr>
<tr><td colspan="7">Susbsistema de I/O del kernel</td></tr>
<tr><td>Driver SCSI</td><td>Driver del teclado</td><td>Driver del mouse</td><td>.....</td><td>Driver del bus PCI</td><td>Driver del diskette</td><td>Driver ATAPI</td></tr>
<tr><td>Controller SCSI</td><td>Controller del teclado</td><td>Controller del mouse</td><td></td><td>Controller del bus PCI</td><td>Controller del diskette</td><td>Controller ATAPI</td></tr>
<tr><td>Device SCSI</td><td>Device del teclado</td><td>Device del mouse</td><td></td><td>Device del bus PCI</td><td>Device del diskette</td><td>Device ATAPI</td></tr>
<tr><td></td><td colspan="6">Hardware</td></tr>
<tr><td></td><td colspan="6">Software</td></tr>
</table>

La capa correspondiente a device drivers esconde al subsistema de I/O del kernel las diferencias entre los diferentes controladores. De la misma manera, las llamadas a sistema (system calls) de I/O son las interfaces entre las aplicaciones y las particularidades del hardware, agrupando éste en unas pocas clases.

Al crear un subsistema de I/O independiente del HW se simplifica la tarea del desarrollador del SO y de los fabricantes del HW.
Consideremos que constantemente se crean nuevos dispositivos de HW y, sin embargo, pueden conectarse rápidamente sin tener que esperar que el desarrollador del SO escriba el código. Esto se logra porque los nuevos dispositivos se adaptan a las interfaces ya existentes.

Diferentes características que tienen los dispositivos

- Orientados a carácter o a bloque
- Acceso secuencial o random
- Sincrónicos o asincronicos
- Compartido o dedicado
- Diferentes velocidades de operación
- Read-Write, Read Only, Write Only

El SO esconde algunas de las características propias de cada dispositivo para facilitar el acceso desde las aplicaciones, agrupando los dispositivos en algunos tipos standard.
El SO provee llamdas a sistema (system calls) especiales para acceder a dispositivos tales como el timer y el reloj (clock) que marca la fecha (date). También para el dispositivo gráfico (graphical display), video y audio.
Las convenciones de acceso incluyen normalmente entrada/salida bloqueante (block I/O), entrada/salida de flujo de caracteres (character-stream I/O), archivo mapeado a memoria (memory mapped file), y sockets de red (network sockets).
La mayoría de los SO proveen llamdas a sistema (system calls) especiales para acceder a los dispositivos desde la aplicación pasándole comandos directamente al controlador de dispositivos (device driver). En el caso de UNIX la system call es ioctl. Con ella se puede acceder a cualquier driver sin tener que crear una nueva system call.
ioctl tiene tres argumentos:
descriptor de archivos: relaciona la aplicación con el driver refiriendose al dispositivo que maneja ese driver.

- Identificador del comando a ejecutar
- Puntero a una estructura de datos en memoria para transferir información de control o dato entre la aplicación y el driver.

Dispositivos de bloque y de carácter

La interface de dispositivo orientado a bloque es la que trata con todos los dispositivos que trabajan en la modalidad de bloque.
El dispositivo entiende comandos como read, write o seek. La aplicación normalmente accede a los dispositivos a través de la interfaz del filesystem (desde la aplicación se utlizan instrucciones del tipo read registro, o read dispositivo).
Cuando el SO o el manejador de base de datos o cualquier aplicación que lo requiera quiere acceder a un dispositivo de bloque como si fuera un arreglo lineal de bloques, lo accede en la modalidad raw I/O (en Unix uno puede en lugar de usar la estructura de filesystem, con inodos y demás, declarar el acceso como raw device, para ver el filesystem como un bloque atrás de otro).

Otros dispositivos son accedidos a través de la interfaz character-stream. Esta interfaz tiene llamadas a sistema básicos del tipo “get character” o “put character”. Tambien se pueden dar acceso por línea permitiendo almacenar y editar dentro de la línea (por ejemplo, usarbackspace).
Este tipo de acceso es el conveniente para teclados, modems, etc

Memory mapped I/O

El controlador de I/O tiene registros donde se mantienen comandos y datos que estan siendo transferidos.
Hay instrucciones de I/O especiales que transfieren los datos entre estos registros y la memoria.
Algunas computadoras usan entrada/salida mapeada a memoria para lograr un acceso mas conveniente. Esto se realiza asignando direcciones de memoria especificas y mapean los registros del dispositivo. La lectura o escritura sobre estas direcciones definen la transferencia hacia y desde los registros del dispositivo.
Por ejemplo se mapea cada ubicación de la pantalla en una direccion de memoria. Poner texto en la pantalla es, simplemente, escribir el texto en el espacio de memoria asignado, es decir en lo que se llama ubicaciones mapeadas a memoria (memory-mapped locations).
En los puertos serial y paralelo es conveniente el uso de entrada/salida mapeada a memoria. En los seriales se conecta, por ejemplo, el modem; en el paralelo, la impresora.
Cuando la CPU desea transferir datos a alguno de estos dispositivos, trabaja en unos registros de dispositivo, un I/O port o puerto de E/S. Si quiere transferir hacia fuera un string de bytes a través del puerto serial, la CPU escribe un byte al registro del dispositivo y enciende un bit de control para indicar que hay información a transferir. El dispositivo toma el byte y establece el bit de control para avisar que el registro está disponible para enviar el próximo byte. Y la CPU puede enviar el próximo.
La forma de la CPU de ver el estado del bit es haciendo escrutinio (polling) entrada/salida programada (programmed I/O, PIO) o recibiendo una interrupción cuando el bit indica que el registro está disponible (interrupt driven).
Algunos SO permiten mapeo a memoria (memory mapping) de archivo, asociando lógicamente una sección de archivo a una parte del espacio de la memoria virtual.
Cada vez que se lee o escribe a esa región es como leer y escribir en el archivo. Al cerrar el archivo en la aplicación se transfiere esa parte de la memoria a la ubicación física real del archivo (en disco, por ejemplo) y se libera el espacio de memoria.
Este método facilita compartir un archivo entre diferentes procesos.
Es común para el SO usar la interfaz de mapping para servicios del kernel. Por ejemplo, para ejecutar un programa, el SO lleva el ejecutable a memoria y transfiere el control a la direccion de comienzo.

Network devices. Sockets

La mayoría de las computadoras ofrecen estos dispositivos cuyas interfaces son diferentes a las de disco, por ejemplo, si bien usan también read, write, seek.
En Unix, la interface que se utiliza es el socket. Cuando una aplicación quiere comunicarse con otra remota, crea un socket a través de una llamada a sistema, para conectarlo a la aplicación remota. Esta aplicación también crea su socket local. Hay system calls para escuchar si alguna aplicación remota se quiere comunicar, para enviar y para recibir paquetes.
Una llamada a sistema, select, permite en aplicaciones servidor, testear si algún socket está esperando un paquete o tiene un paquete para enviar. De esta manera se evita el escrutinio (polling) sobre el dispositivo.

I/O bloqueante y no bloqueante

Cuando una aplicación emite una llamada a sistema bloqueante la aplicación se suspende y pasa del estado de ejecución (running) al de waiting, pasando a la cola de espera. Cuando se completa la E/S pasa a la cola de listos.
La razón para usar blocking es que no podemos determinar exactamente cuánto va a tardar una E/S.
Pero algunos procesos a nivel de usuario necesitan usar entrada/salida no bloqueante (nonblocking I/O – E/S asincrónica). Tal es el caso del input por teclado y por mouse. O una aplicación de video que lee marcos desde un disco, mientras va descomprimiendo y mostrando el output en la pantalla.
Para poder seguir procesando mientras se realiza I/O, se utilizan aplicaciones multihilo (multithread).
Otra alternativa es usar system calls asincrónicos. Un system call de este tipo retorna enseguida sin esperar que se complete la E/S. Más tarde, a través de un seteo de una variable en su espacio de direcciones, o por una interrupción por software a la aplicación, se la avisa que se completó la E/S.
Veamos la diferencia entre nonblocking y llamadas a sistema asincrónicos: un read nonblocking retorna sin esperar que el dato esté disponible mientras que el system call asincronico espera que se realice la transferencia pero no que se complete la operación.

El subsistema de I/O del kernel

Los servicios que provee el kernel para la I/O son: planificación de la E/S (I/O scheduling), buffering, caching, spooling, reserva de dispositivo y error handling (manejo de errores).
I/O scheduling: es una manera a través de la cual se mejora la eficiencia de una E/S. Otra manera es con técnicas que permitan almacenar en memoria principal o en disco, como ocurre con buffering, caching, etc.
Realizar la planificación consiste en determinar un orden para la ejecución de las distintas operaciones de E/S. Hay que considerar que el scheduling debe mejorar la performance del sistema compartiendo los dispositivos y reduciendo el tiempo de espera para que se complete la E/S.
Buffering: es un concepto que ya hemos visto. Recordemos que es un área de memoria que almacena datos mientras ellos son transferidos entre dos dispositivos, o un dispositivo y una aplicación.
Caching: también es un concepto que conocemos. Es una región de memoria rápida que mantiene copias de datos. Es más eficiente el acceso a esta copia que al original.
Spooling: el spool es un buffer que mantiene la salida para un dispositivo, como por ejemplo una impresora. Es útil por ejemplo cuando varias aplicaciones quieren imprimir a la vez sobre una impresora.
Error handling: Los dispositivos y las transferencias de E/S pueden fallar. El SO a menudo tiene que compensar estas fallas, por ejemplo, rehaciendo un read, o un resend.

Entrada- salida: estructuras en memoria secundaria

Estructura de disco

Podemos ver los discos como una sucesión de bloques lógicos, siendo este bloque la unidad mínima de transferencia. La medida habitual de los bloques es de 512 bytes, pero también se utilizan bloques de 1024, o más.

Los sectores del disco forman pistas (tracks) y estas pistas, cilindros. Los bloques se mapean sobre esta estructura. El sector 0 (cero) es el primer sector, del primer track del cilindro mas externo. Por lo tanto el primer bloque lógico estará formado por ese sector y los sucesivos dentro del primer track del primer cilindro.
Por eso toda dirección de bloque podemos llevarla a la vieja dirección (cilindro, pista, sector). Pero la vieja modalidad tiene sus inconvenientes: hay sectores en mal estado y el numero de sectores por pista no es constante.
Hoy los discos usan “zonas de cilindros”, donde el número de sectores por pista son constantes dentro de la zona.
La tecnología ha permitido aumentar la cantidad de sectores por pistas y la cantidad de cilindros por disco.
Hablemos ahora sobre la velocidad del disco. La mayoría de los drives rotan entre 80 y 150 veces por segundo.
El tiempo de transferencia es la velocidad a la que el dato pasa desde el drive a la computadora.
El tiempo de posicionamiento o de acceso (access time) es el tiempo que tarda el brazo del disco en moverse al cilindro deseado (seek time) sumado al tiempo que tarda el sector deseado en llegar a la cabeza del brazo en la rotación (rotational latency). Hoy los discos transfieren varios megabytes por segundo. Los tiempos vistos (seek y rotational latency) están en el orden de varios milisegundos, pues el tiempo de acceso resultante debe ser rápido.
El ancho de banda en el disco (disk bandwidth) es el número del total de bytes que se transfieren dividido por el tiempo total entre el inicio del requerimiento de servicio hasta que se completa la transferencia.
La cabeza no tiene contacto con la superficie (flota sobre un colchón de aire del orden de los micrones) cuando la cabeza toca la superficie, la daña y ocurre un “aterrizaje de cabezas” o head crash.
La optimización del manejo de la E/S se basa en lograr un buen tiempo de acceso sumado a una buena administración de los requerimientos de E/S a disco. Planificación del uso de disco (Disk Scheduling)
Cuando un proceso necesita hacer una I/O hacia o desde el disco utiliza system calls, donde se especifica si es una entrada o una salida, la dirección de disco a transferir, la dirección de memoria a donde transferir y la cantidad de bytes a transferir.
Cuando el disk drive o el controlador están ocupados, los requerimientos se colocan en una cola dependientes a ese drive.

Distintos tipos de algoritmos de scheduling

Para analizar los distintos tipos de planificación consideraremos una serie de referencias a cilindros donde están los bloques que se quieren acceder.
Sea la siguiente serie, en este orden (cabe aclarar que estamos representando los requerimientos de acceso a cilindros de disco, por solicitudes de I/O).
85, 138, 42, 125, 15, 140, 87, 90 y que actualmente la cabeza lectora-grabadora esta en el cilindro 68.

FCFS

Es la forma más simple de planificación de disco. En nuestro ejemplo, la cabeza se moverá del 68 al 85, de ahí al 138, luego al 42, al 125, al 15, al 140, al 87, al 90.
Este pasaje totaliza 540 movimientos de la cabeza sobre el disco.
La cabeza del disco va de un extremo al otro (del 42 al 125, y luego vuelve al 15, por ejemplo).

SSTF

Podemos mejorar el FCFS pensando en ordenar los acceso de manera tal de acceder a los que están más cercanos entre si primero. Así funciona el algoritmo shortest-seek-timefirst (SSTF) primero el de tiempo de posicionamiento más corto.
Se trata de elegir el próximo movimiento según el menor seek time desde donde está la cabeza del disco.
Consideremos que el seek time aumenta cuando más cilindros atraviesa la cabeza para llegar al destino.
En nuestro ejemplo, el trayecto sería: 68, 85, 87, 90, 125, 138, 40 , 42 o sea, en total, 170 movimientos de cabeza.
El problema es que puede provocar starvation (inanición) pues un requerimiento a un cilindro cercano a los extremos puede quedar siempre postergado por la llegada de requerimientos nuevos cercanos a la ubicación en ese momento de la cabeza (es probable que se den un conjunto de requerimientos de acceso a cilindros cercanos entre sí). El SSTF introduce una mejora con respecto al FCFS, pero no es óptimo.

SCAN

En este algoritmo la cabeza se posiciona en un extremo del disco y va avanzando hacia el otro extremo, atendiendo los requerimientos de acceso a cilindro que va encontrando por el camino.
También se le llama “el algoritmo del ascensor” pues su comportamiento es semejante al que realiza un ascensor automático: a medida que va subiendo o bajando va atendiendo las llamadas en orden.
Veamos en nuestra serie cómo actuaría. En primer lugar se debe analizar hacia que extremo se moverá (del 68 para el 67 o para el 69?).
Resultaría:
15, 42, 85, 87, 90, 125, 138, 140 Son 178 movimientos.
Consideremos que, a medida que la cabeza va avanzando, van llegando requerimientos de acceso a cilindros que ya pasaron y deben esperar que la cabeza vuelva luego de llegar al extremo hacia el que se dirige.
Cuando llega al extremo, al volver encuentra inicialmente pocos requerimientos (recién fue atendida esa zona). La mayor cantidad de requerimientos pendientes estarán en el extremo opuesto.

C-SCAN

Para mantener uniforme el tiempo de espera de los requerimientos, se modifica el SCAN y se usa el C-SCAN.
En este algoritmo la cabeza, al llegar al extremo final se vuelve a posicionar en el inicio, para atender primero los requerimientos que, se supone por lógica, hace mas tiempo que están esperando.

LOOK

En este algoritmo lo que se modifica es que la cabeza no llega hasta el extremo del disco, sino hasta el requerimiento más cercano al final, y desde allí retorna.
Por lo tanto podemos verlo como modificaciones al SCAN y C-SCAN, que, en este caso, pasan a llamarse LOOK y C-LOOK.

Cómo se selecciona el algoritmo a utilizar

Dada cada serie de referencias se podría calcular el acceso óptimo pero ello conduciría a procesar cada serie, lo que llevaría tiempo que no justifica la optimización.
El SSTF es el de uso mas común. Para sistemas de mucha carga de disco, el SCAN y CSCAN son útiles (hay menos posibilidad de inanición).
Uno de los puntos que influye en el orden que se encolan los requerimientos son los métodos de asignación de archivos que se usen. Si es un archivo de asignación contigua los requerimientos de acceso a los cilindros serán cercanos. No será si, en cambio, en el caso de un archivo linkeado o indexado.
Donde se ubican dentro del disco aquellos bloques que son muy accedidos (como directorios o bloques de indireccion) se podrían tratar de mantener en memoria el mayor tiempo posible para ganar tiempo de acceso.
Normalmente se eligen SSTF o LOOK como algoritmo default. No obstante seria importante que el algoritmo fuera un modulo separado del SO que pueda ser reemplazado si es necesario.
Algunos fabricantes de discos hoy incluyen algunas características para planificación en el HW del controlador.
Cuando el SO manda una serie de referencias el controller las encola y planifica para mejorar los tiempos de acceso. Pero no olvidemos que alguna de estas referencias puede tener una prioridad asociada: si es para recuperar una página en administración de memoria paginada o un segmento, debería acceder antes que un requerimiento a un archivo por un usuario no privilegiado.
También para mantener íntegra la estructura de file system puede optarse por priorizar algunos accesos. Si al crear un archivo se empieza a mandar datos antes de que quede modificada la lista de inodos o donde se marca el espacio libre en disco, puede atentar contra la robustez del filesystem.

Otras características de administración de disco

Formateo de disco

Antes de usar un disco nuevo hay que dividirlo en sectores que el controlador pueda leer y escribir. Es lo que se llama formateo de bajo nivel o físico.
A cada sector se le da una estructura que tiene un header (encabezamiento), un área de datos (normalmente 512 bytes) y un trailer (cola, final).
El header y el trailer tienen información de control para el controlador como número. de sector y un ECC que es un código para detectar errores. Este código sirve para determinar si los datos fueron corrompidos.
Cuando se lee el sector, se calcula el ECC con los bits que lo componen y se compara con el que esta almacenado. Si es igual, entonces el sector no fue alterado desde su escritura.
Cuando se modifica un sector, se calcula el ECC y se lo almacena dentro del sector.
Para poder almacenar archivos, el SO necesita registrar sus propias estructuras de datos en el disco. Por un lado, debe particionar el disco, donde cada partición contiene uno o mas grupos de cilindros. Luego puede usar cada partición incluso como un disco separado. Puede optarse en tener en una partición todo el código ejecutable del SO y en otra, todo el filesystem. O un SO en una partición, por ejemplo Windows, y en otra, otro SO (LINUX, por ejemplo).

Luego de particionar, para poder almacenar archivos, hay que construir una estructura de filesystem (formateo lógico).
Se trata de almacenar las estructuras iniciales como la FAT o los inodos UNIX, directorios iniciales vacíos, etc.
En algunos casos puedo usar la partición como si fuera un arreglo de bloques lógicos sin formato de filesystem (caso de BD grandes) y entonces le damos a la partición un formato especial (raw I/O).

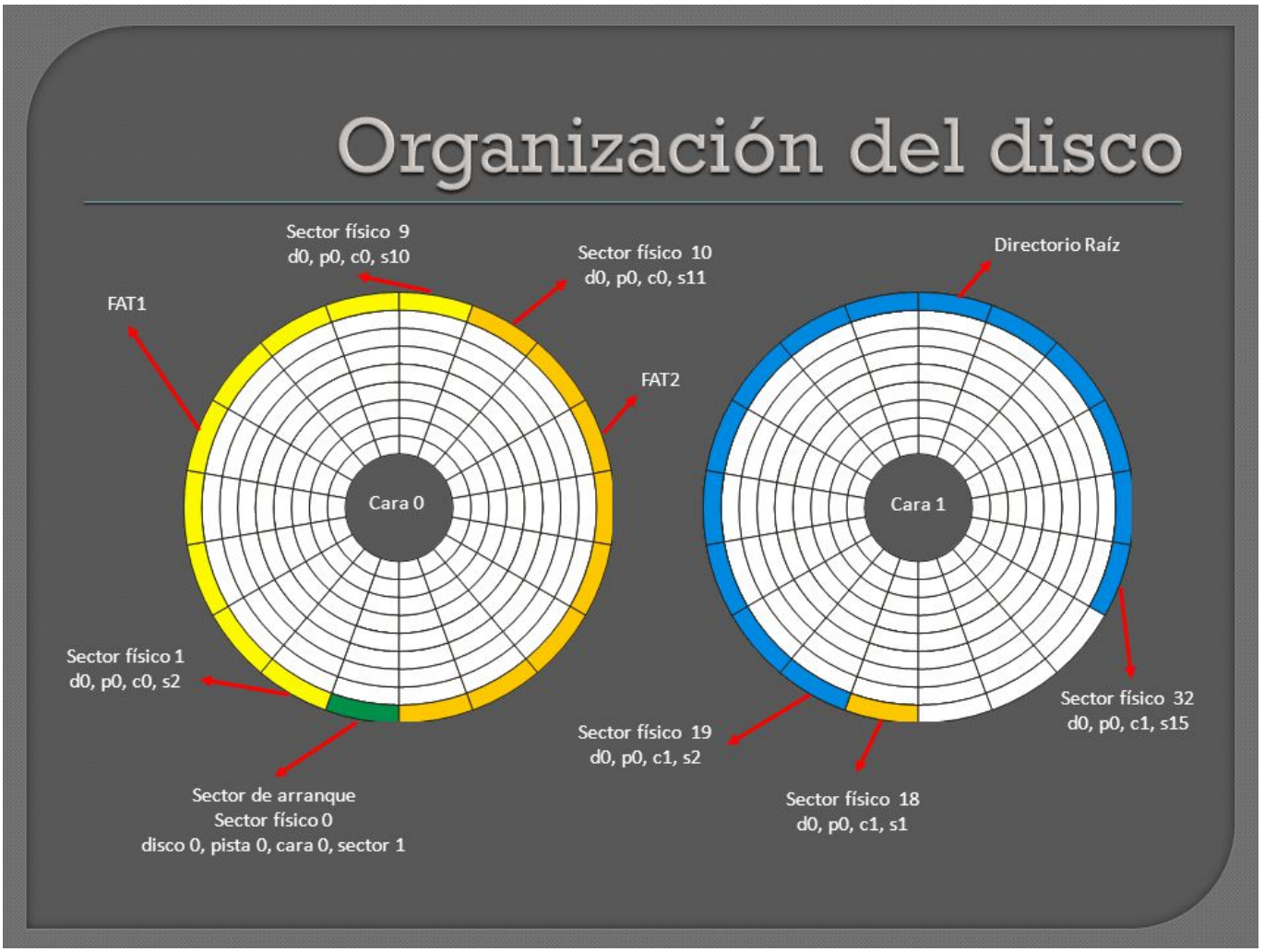
Facultad de Ingeniería
marcelitacamen1@gmail.com

Sector de Arranque

Es el primer sector de todo disco duro (cabeza 0, cilindro 0, sector 1). En él se almacena la tabla de particiones y un pequeño programa master de inicialización, llamado también MasterBoot.

Este programa es el encargado de leer la tabla de particiones y ceder el control al sector de arranque de la partición activa.


ORGANIZACION DE LA INFORMACION EN UN DISCO FLEXIBLE
Pista 0 (Track)
Sector 1
Pista 0,Sector 1
Sector de arranque (Boot)
Tabla de LOcalizacion de Archivos (FAT)
Directorio Raiz
Area de Datos
DISCO DE 3 1/2 Y 1.44 MEGAS



Boot Block

Cuando se enciende una computadora es necesario inicializar el sistema: registros de CPU, preparar los disk controllers, contenidos de la memoria principal, arrancar el SO.
Esto se realiza a través del programa bootstrap.
Este programa, muy simple, se encarga de buscar el kernel del SO en el disco, cargarlo en memoria y saltar a la dirección de inicio para comenzar la ejecución.
Reside en una ROM (read only memory) que al ser solo de lectura no puede ser afectada por virus. Para cambiar esta ROM es necesario cambiar el chip (HW) que la contiene.
Por eso normalmente no se le da mas funcionalidad que la de carga, desde el disco del SO.
De allí que podamos instalar otra versión de SO sin modificar el HW.
En el chip está un pequeño bootstrap. El completo está en disco, en una partición que se llama bootblock, en un lugar fijo. El disco que contiene esta partición es llamado boot disk o system disk.

CD-ROMs

En años recientes se han empezado a utilizar los discos ópticos (en contraste a los magnéticos). Estos discos tienen densidades de grabación mucho más altas que los discos magnéticos convencionales. Los discos ópticos se desarrollaron en un principio para grabar programas de televisión, pero se les puede dar un uso más estético como dispositivos de almacenamiento de computadora. Debido a su capacidad potencialmente enorme, los discos ópticos han sido tema de una gran cantidad de investigación y han pasado por una evolución increíblemente rápida.
Los discos ópticos de primera generación fueron inventados por el conglomerado de electrónica holandés Philips, para contener películas. Tenían 30 centímetros de diámetro y se comercializaron bajo el nombre LaserVision, pero no tuvieron mucha popularidad, excepto en Japón.
En 1980, Philips y Sony desarrollaron el CD (Disco Compacto), que sustituyó rápidamente al disco de vinilo de 33 1/3 RPM que se utilizaba para música (excepto entre los conocedores, que aún preferían el vinilo). Los detalles técnicos precisos para el CD se publicaron en un Estándar Internacional oficial (IS 10149) conocido popularmente como el **Libro rojo** debido al color de su portada (los Estándares Internacionales son emitidos por la Organización Internacional de Estándares, que es la contraparte internacional de los grupos de estándares nacionales como ANSI, DIN, etc. Cada uno tiene un número IS). El punto de publicar las especificaciones de los discos y las unidades como un estándar internacional tiene como fin permitir que los CDs de distintas compañías disqueras y los reproductores de distintos fabricantes electrónicos puedan funcionar en conjunto. Todos los CDs tienen 120 mm de diámetro y 1.2 mm de grosor, con un hoyo de 15 mm en medio. El CD de audio fue el primer medio de almacenamiento digital masivo en el mercado. Se supone que deben durar 100 años. Por favor consulte de nuevo en el 2080 para saber cómo le fue al primer lote.
Un CD se prepara en varios pasos. El primero consiste en utilizar un láser infrarrojo de alto po- der para quemar hoyos de 0.8 micrones de diámetro en un disco maestro con cubierta de vidrio. A partir de este disco maestro se fabrica un molde, con protuberancias en lugar de los hoyos del láser. En este molde se inyecta resina de policarbonato fundido para formar un CD con el mismo patrón de hoyos que el disco maestro de vidrio. Después se deposita una capa muy delgada de aluminio reflectivo en el policarbonato, cubierta por una laca protectora y finalmente una etiqueta. Las de- presiones en el sustrato de policarbonato se llaman **hoyos** (*pits*); las áreas no quemadas entre los hoyos se llaman **áreas lisas** (*lands*).

Cuando se reproduce, un diodo láser de baja energía emite luz infrarroja con una longitud de onda de 0.78 micrones sobre los hoyos y áreas lisas a medida que van pasando. El láser está del lado del policarbonato, por lo que los hoyos salen hacia el láser como protuberancias en la superficie de área lisa. Como los hoyos tienen una altura de un cuarto de la longitud de onda de la luz del láser, la luz que se refleja de un hoyo está desfasada por media longitud de onda con la luz que se refleja de la superficie circundante. Como resultado, las dos partes interfieren en forma destructiva y devuelven me- nos luz al fotodetector del reproductor que la luz que rebota de un área lisa. Así es como el reproductor puede diferenciar un hoyo de un área lisa. Aunque podría parecer más simple utilizar un hoyo para grabar un 0 y un área lisa para grabar un 1, es más confiable utilizar una transición de hoyo a área li- sa o de área lisa a un hoyo para un 1 y su ausencia como un 0, por lo que se utiliza este esquema.
Los hoyos y las áreas lisas se escriben en una sola espiral continua, que empieza cerca del hoyo y recorre una distancia de 32 mm hacia el borde. La espiral realiza 22,188 revoluciones alrededor del

disco (aproximadamente 600 por milímetro). Si se desenredara, tendría 5.6 km de largo. La espiral se ilustra en la figura 5-21.

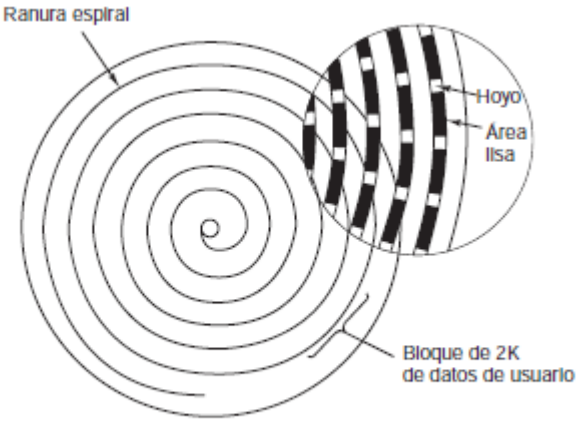


Figura 5-21. Estructura de grabación de un disco compacto o CD-ROM.

Para reproducir la música a una velocidad uniforme, es necesario un flujo continuo de hoyos y áreas a una velocidad lineal constante. En consecuencia, la velocidad de rotación del CD se debe reducir en forma continua a medida que la cabeza de lectura se desplaza desde la parte interna del CD hacia la parte externa. En el interior la velocidad de rotación es de 530 RPM para lograr la velocidad de flujo continuo deseada de 120 cm/seg; en el exterior tiene que reducirse a 200 RMP para proporcionar la misma velocidad lineal en la cabeza. Una unidad de velocidad lineal constante es muy distinta a una unidad de disco magnético, que opera a una velocidad angular constante, sin importar dónde se encuentre la cabeza en un momento dado. Además, 530 RPM están muy lejos de las 3600 a 7200 RPM a las que giran la mayoría de los discos magnéticos.

En 1984, Philips y Sony se dieron cuenta del potencial de utilizar CDs para almacenar datos de computadora, por lo cual publicaron el **Libro amarillo** que define un estándar preciso para lo que se conoce ahora como **CD-ROM**s (*Compact disk-read only memory,* Disco compacto–memoria de sólo lectura). Para apoyarse en el mercado de CDs de audio, que para ese entonces ya era considerable, los CD-ROMs tenían que ser del mismo tamaño físico que los CDs de audio, ser compatibles en sentido mecánico y óptico con ellos, y se debían producir utilizando las mismas máquinas de moldeado por inyección de policarbonato. Las consecuencias de esta decisión fueron que no sólo se requerían motores lentos de velocidad variable, sino también que el costo de fabricación de un CD-ROM estaría muy por debajo de un dólar en un volumen moderado.

Lo que definió el Libro amarillo fue el formato de los datos de computadora. También mejoró las habilidades de corrección de errores del sistema, un paso esencial pues aunque a los amantes de la música no les importaba perder un poco aquí y allá, los amantes de las computadoras tendían a ser Muy Exigentes en cuanto a eso. El formato básico de un CD-ROM consiste en codificar cada byte en un símbolo de 14 bits, que es suficiente como para usar el código de Hamming en un byte de 8 bits con 2 bits de sobra. De hecho, se utiliza un sistema de codificación más potente. La asignación de 14 a 8 para la lectura se realiza en el hardware mediante la búsqueda en una tabla.

En el siguiente nivel hacia arriba, un grupo de 42 símbolos consecutivos forma una **estructura** de 588 bits. Cada estructura contiene 192 bits de datos (24 bytes). Los 396 bits restantes se utilizan para corrección y control de errores. De éstos, 252 son los bits de corrección de errores en los símbolos de 14 bits, y 144 se llevan en las transmisiones de símbolos de 8 bits. Hasta ahora, este esquema es idéntico para los CDs de audio y los CD-ROMs.

Lo que agrega el Libro amarillo es el agrupamiento de 98 estructuras en un **sector de CD-ROM**, como se muestra en la figura 5-22. Cada sector del CD-ROM empieza con un preámbulo de 16 bytes, del cual los primeros 12 son 00FFFFFFFFFFFFFFFFFFFF00 (hexadecimal) para permitir que el reproductor reconozca el inicio de un sector de CD-ROM. Los siguientes 3 bytes contienen el número

de sector, que se requiere debido a que es mucho más difícil realizar búsquedas en un CD-ROM con una sola espiral de datos que en un disco magnético, con sus pistas concéntricas uniformes. Para buscar, el software en la unidad calcula aproximadamente a dónde debe ir, desplaza ahí la cabeza y después empieza a buscar un preámbulo para ver qué tan buena fue su aproximación. El último byte del preámbulo es el modo.

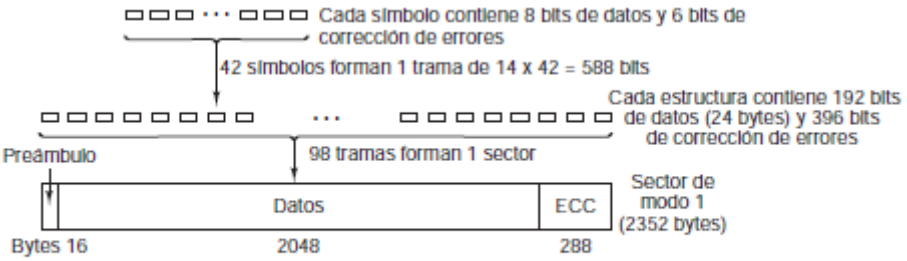


Figura 5-22. Distribución lógica de los datos en un CD-ROM.

El Libro amarillo define dos modos. El modo 1 utiliza la distribución de la figura 5-22. Con un preámbulo de 16 bytes, 2048 bytes de datos y un código de corrección de errores de 288 bytes

(un código Reed-Solomon cruzado entrelazado). El modo 2 combina los campos de datos y ECC en un campo de datos de 2336 bytes para aquellas aplicaciones que no necesitan (o que no pueden darse el tiempo de llevar a cabo) la corrección de errores, como el audio y el video. Hay que tener en cuenta que para ofrecer una excelente confiabilidad, se utilizan tres esquemas separados de corrección de errores: dentro de un símbolo, dentro de una estructura y dentro de un sector del CD-ROM. Los errores de un solo bit se corrigen en el nivel más bajo, los errores de ráfagas cortas se corrigen en el nivel de estructura y los errores residuales se atrapan en el nivel de sector. El precio a pagar por esta confiabilidad es que se requieren 98 estructuras de 588 bits (7203 bytes) para llevar a cabo una sola transferencia de 2048 bytes, una eficiencia de sólo 28%.

Las unidades de CD-ROM de una sola velocidad operan a 75 sectores/seg, con lo cual se obtiene una velocidad de transferencia de datos de 153,600 bytes/seg en modo 1 y de 175,200 by- tes/seg en modo 2. Las unidades de doble velocidad son el doble de rápidas, y así en lo sucesivo hasta la velocidad más alta. Por ende, una unidad de 40x puede transferir datos a una velocidad de 40 × 153,600 bytes/seg, suponiendo que la interfaz de la unidad, el bus y el sistema operativo puedan manejar esta velocidad de transferencia de datos. Un CD de audio estándar tiene espacio para 74 minutos de música que, si se utilizan para datos en modo 1, logran una capacidad de 681,984,000 bytes. Esta cifra se reporta comúnmente como 650 MB, debido a que 1 MB equivale a 2^{20} bytes (1,048,576 bytes) y no a 1,000,000 bytes.

Tenga en cuenta que ni siquiera una unidad de CD-ROM de 32x (4,915,200 bytes/s) le hace frente a una unidad de disco magnético SCSI rápida a 10 MB/seg, aunque muchas unidades de CD-ROM utilizan la interfaz SCSI (también existen las unidades IDE de CD-ROM). Cuando se da uno cuenta de que el tiempo de búsqueda generalmente es de varios cientos de milisegundos, queda claro que las unidades de CD-ROM no están en la misma categoría de rendimiento que las unidades de disco magnético, a pesar de su gran capacidad.

En 1986, Philips publicó el **Libro verde**, agregando gráficos y la habilidad de entrelazar audio, video y datos en el mismo sector, una característica esencial para los CD-ROMs de multimedia.

La última pieza del rompecabezas del CD-ROM es el sistema de archivos. Para que fuera posible usar el mismo CD-ROM en distintas computadoras, era necesario un acuerdo sobre los sistemas de archivos de CD-ROM. Para llegar a este acuerdo, se reunieron los representantes de muchas compañías de computadoras en Lake Tahoe, en la región High Sierra del límite entre California y

Nevada, e idearon un sistema de archivos al que llamaron **High Sierra**. Después evolucionó en un Estándar Internacional (IS 9660). Tiene tres niveles. El nivel 1 utiliza nombres de archivo de hasta 8 caracteres, seguidos opcionalmente de una extensión de hasta 3 caracteres (la convención de denominación de archivos de MS-DOS). Los nombres de archivo pueden contener sólo letras mayúsculas, dígitos y el guión bajo. Los directorios se pueden anidar hasta ocho niveles, pero los nombres de los directorios no pueden contener extensiones. El nivel 1 requiere que todos los archivos sean contiguos, lo cual no es problema en un medio en el que se escribe sólo una vez. Cualquier CD-ROM que cumpla con el nivel 1 del IS 9660 se puede leer utilizando MS-DOS, una computadora Apple, una computadora UNIX o casi cualquier otra computadora. Los editores de CD-ROMs consideran que esta propiedad es una gran ventaja.
El nivel 2 del IS 9660 permite nombres de hasta 32 caracteres, y el nivel 3 permite archivos no contiguos. Las extensiones Rock Ridge (denominadas en honor a la ciudad en la que se desarrolla el filme de Gene Wilder llamado *Blazing Saddles*) permiten nombres muy largos (para UNIX),

UIDs, GIDs y vínculos simbólicos, pero los CD-ROMs que no cumplan con el nivel 1 no podrán leerse en todas las computadoras.

Los CD-ROMs se han vuelto en extremo populares para publicar juegos, películas, enciclopedias, atlas y trabajos de referencia de todo tipo. La mayoría del software comercial viene ahora en CD-ROM. Su combinación de gran capacidad y bajo costo de fabricación los hace adecuados para innumerables aplicaciones.

CD-Grabables

En un principio, el equipo necesario para producir un CD-ROM maestro (o CD de audio, para esa cuestión) era extremadamente costoso. Pero como siempre en la industria de las computadoras, nada permanece costoso por mucho tiempo. A mediados de la década de 1990, los grabadores de CDs no más grandes que un reproductor de CD eran un periférico común disponible en la mayoría de las tiendas de computadoras. Estos dispositivos seguían siendo distintos de los discos magnéticos, porque una vez que se escribía información en ellos no podía borrarse. Sin embargo, rápidamente encontraron un nicho como medio de respaldo para discos duros grandes y también permitieron que individuos o empresas que iniciaban operaciones fabricaran sus propios CD-ROMs de distribución limitada, o crear CDs maestros para entregarlos a plantas de duplicación de CDs comerciales de al- to volumen. Estas unidades se conocen como **CD-R**s (**CD-Grabables**).
Físicamente, los CD-Rs empiezan con discos en blanco de policarbonato de 120 mm, que son similares a los CD-ROMs, excepto porque contienen una ranura de 0.6 mm de ancho para guiar el láser para la escritura. La ranura tiene una excursión sinoidal de 0.3 mm a una frecuencia exacta de 22.5 kHz para proveer una retroalimentación continua, de manera que la velocidad de rotación se pueda supervisar y ajustar con precisión, en caso de que sea necesario. Los CD-Rs tienen una apariencia similar a los CD-ROMs, excepto porque tienen una parte superior dorada, en lugar de una plateada. El color dorado proviene del uso de verdadero oro en vez de aluminio para la capa reflectora. A diferencia de los CDs plateados que contienen depresiones físicas, en los CD-Rs la distinta reflectividad de hoyos y áreas lisas se tiene que simular. Para ello hay que agregar una capa de colorante entre el policarbonato y la capa de oro reflectiva, como se muestra en la figura 5-23. Se utilizan dos tipos de colorante: la cianina, que es verde, y la ptalocianina, que es de color naranja con amarillo. Los químicos pueden argumentar indefinidamente sobre cuál es mejor. Estos colorantes son similares a los que se utilizan en la fotografía, lo cual explica por qué Eastman Kodak y Fuji son los principales fabricantes de CD-Rs en blanco.

En su estado inicial, la capa de colorante es transparente y permite que la luz del láser pase a través de ella y se refleje en la capa dorada. Para escribir, el láser del CD-R se pone en alto poder (8 a 16 mW). Cuando el haz golpea en un punto del colorante, se calienta y quebranta un lazo químico. Este cambio en la estructura molecular crea un punto oscuro. Cuando se lee de vuelta (a 0.5 mW), el fotodetector ve una diferencia entre los puntos oscuros en donde se ha golpeado el colorante y las áreas transparentes donde está intacto. Esta diferencia se interpreta como la diferencia entre los hoyos y las áreas lisas, aun y cuando se lea nuevamente en un lector de CD-ROM regular, o incluso en un reproductor de CD de audio.
Ningún nuevo tipo de CD podría andar con la frente en alto sin un libro de colores, por lo que el CD-R tiene el **Libro naranja**, publicado en 1989. Este documento define el CD-R y también un

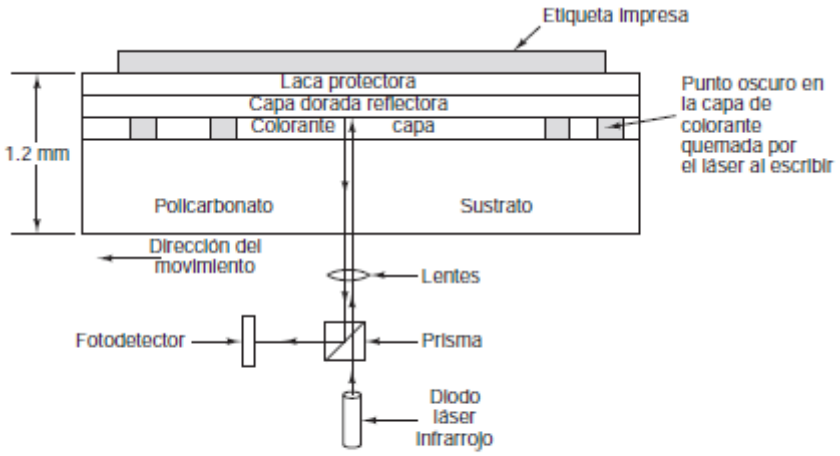


Figura 5-23. Sección transversal de un disco CD-R y el láser (no está a escala). Un CD-ROM plateado tiene una estructura similar, excepto sin la capa de colorante y con una capa de aluminio picada en vez de una capa dorada.

nuevo formato, el **CD-ROM XA**, que permite escribir CD-Rs en forma incremental; unos cuantos sectores hoy, otros pocos mañana, y unos cuantos el siguiente mes. A un grupo de sectores consecutivos que se escriben a la vez se le llama **pista de CD-ROM**.

Uno de los primeros usos del CD-R fue para el PhotoCD de Kodak. En este sistema, el cliente trae un rollo de película expuesta y su viejo PhotoCD al procesador de fotografías, y obtiene de vuelta el mismo PhotoCD, al que se le agregan las nuevas fotografías después de las anteriores. El nuevo lote, que se crea al explorar los negativos, se escribe en el PhotoCD como una pista separada en el CD-ROM. Se requería la escritura incremental debido a que cuando se introdujo este pro- ducto, los CD-R en blanco eran demasiado costosos como para ofrecer uno nuevo por cada rollo de película.
Sin embargo, la escritura incremental crea otro problema. Antes del Libro naranja, todos los CD-ROMs tenían una sola **VTOC** (*Volume Table of Contents,* Tabla de contenido del volumen) al principio. Ese esquema no funciona con las escrituras incrementales (es decir de varias pistas). La solución del Libro naranja es proporcionar a cada pista de CD-ROM su propia VTOC. Los archivos que se listan en la VTOC pueden incluir algunos o todos los archivos de las pistas anteriores. Una vez que se inserta el CD-R en la unidad, el sistema operativo busca a través de todas las pistas de CD-ROM para localizar la VTOC más reciente, que proporciona el estado actual del disco. Al incluir algunos, pero no todos los archivos de las pistas anteriores en la VTOC actual, es posible dar la ilusión de que se han eliminado archivos. Las pistas se pueden agrupar en **sesiones**, lo cual conlleva a los CD-ROMs de **multisesión**. Los reproductores de CDs de audio estándar no pueden manejar los CDs de multisesión, ya que esperan una sola VTOC al principio. Sin embargo, algunas aplicaciones de computadora pueden manejarlos.

El CD-R hace posible que los individuos y las empresas copien fácilmente CD-ROMs (y CDs de audio), por lo general violando los derechos del editor. Se han ideado varios esquemas para dificultar la piratería y para que sea difícil leer un CD-ROM utilizando otra cosa que no sea el software del editor. Uno de ellos implica grabar todas las longitudes de los archivos en el CD-ROM como de varios gigabytes, para frustrar cualquier intento de copiar los archivos en disco duro mediante el uso de software de copiado estándar. Las verdaderas longitudes se incrustan en el software del editor o se ocultan (posiblemente cifradas) en el CD-ROM, en un lugar inesperado. Otro esquema utiliza ECCs incorrectos de manera intencional en sectores seleccionados, esperando que el software de copiado de CDs “corrija” esos errores. El software de aplicación comprueba los ECCs por sí mismo, rehusándose a funcionar si están correctos. El uso de huecos no estándar entre las pistas y otros “defectos” físicos también es posible.

CD-Regrabables

Aunque las personas están acostumbradas a otros medios de escritura de sólo una vez como el papel y la película fotográfica, hay una demanda por el CD-ROM regrabable. Una tecnología que ahora está disponible es la del **CD-RW** (**CD-Regrabable**), que utiliza medios del mismo tamaño que el CD-R. Sin embargo, en vez de colorante de cianina o ptalocianina, el CD-RW utiliza una aleación de plata, indio, antimonio y telurio para la capa de grabación. Esta aleación tiene dos estados estables: cristalino y amorfo, con distintas reflectividades.

Las unidades de CD-RW utilizan láseres con tres potencias: en la posición de alta energía, el láser funde la aleación y la convierte del estado cristalino de alta reflectividad al estado amorfo de baja reflectividad para representar un hoyo; en la posición de energía media, la aleación se funde y se vuelve a formar en su estado cristalino natural para convertirse en un área lisa nuevamente; en baja energía se detecta el estado del material (para la lectura), pero no ocurre una transición de estado.

La razón por la que el CD-RW no ha sustituido al CD-R es que los CD-RW en blanco son más costosos. Además, para las aplicaciones que consisten en respaldar discos duros, el hecho de que una vez escrito el CD-R no se pueda borrar accidentalmente es una gran ventaja.

DVD

El formato básico de CD/CD-ROM ha estado en uso desde 1980. La tecnología ha mejorado desde entonces, por lo que ahora los discos ópticos de mayor capacidad son económicamente viables y hay una gran demanda por ellos. Hollywood estaría encantado de eliminar las cintas de video análogas a favor de los discos digitales, ya que los discos tienen una mayor calidad, son más económicos de fabricar, duran más tiempo, ocupan menos espacio en las repisas de las tiendas de video y no tienen que rebobinarse. Las empresas de electrónica para el consumidor siempre están buscan- do un nuevo producto que tenga un gran éxito, y muchas empresas de computadoras desean agregar características de multimedia a su software.

Esta combinación de tecnología y demanda por tres industrias inmensamente ricas y poderosas conllevó al **DVD**, que originalmente era un acrónimo para **Video disco digital (***Digital Video Disk***)**,

pero que ahora se conoce oficialmente como **Disco versátil digital (***Digital Versatile Disk***)**. Los DVDs utilizan el mismo diseño general que los CDs, con discos de policarbonato moldeado por inyección de 120 mm que contienen hoyos y áreas lisas, que se iluminan mediante un diodo láser y se leen mediante un fotodetector. Lo nuevo es el uso de

1. Hoyos más pequeños (0.4 micrones, en comparación con 0.8 micrones para los CDs).
2. Una espiral más estrecha (0.74 micrones entre pistas, en comparación con 1.6 micrones para los CDs).
3. Un láser rojo (a 0.65 micrones, en comparación con 0.78 micrones para los CDs).

En conjunto, estas mejoras elevan la capacidad siete veces, hasta 4.7 GB. Una unidad de DVD 1x opera a 1.4 MB/seg (en comparación con los 150 KB/seg de los CDs). Por desgracia, el cambio a los láseres rojos utilizados en los supermercados implica que los reproductores de DVD requieren un segundo láser o una óptica compleja de conversión para poder leer los CDs y CD-ROMs existentes. Pero con la disminución en el precio de los láseres, la mayoría de los reproductores de DVD tienen ahora ambos tipos de láser para leer ambos tipos de medios.
¿Es 4.7 GB suficiente? Tal vez. Mediante el uso de la compresión MPEG-2 (estandarizada en el IS 13346), un disco DVD de 4.7 GB puede contener 133 minutos de video de pantalla y movi- miento completo en alta resolución (720 x 480), así como pistas de sonido en hasta ocho lenguajes y subtítulos en 32 más. Cerca de 92% de todas las películas que se hayan realizado en Hollywood son de menos de 133 minutos. Sin embargo, algunas aplicaciones como los juegos multimedia o las obras de consulta pueden requerir más, y a Hollywood le gustaría colocar varias películas en el mismo disco, por lo que se han definido cuatro formatos:

1. Un solo lado, una sola capa (4.7 GB).
2. Un solo lado, doble capa (8.5 GB).
3. Doble lado, una sola capa (9.4 GB).
4. Doble lado, doble capa (17 GB).

¿Por qué tantos formatos? En una palabra: política. Philips y Sony querían discos de un so- lo lado, doble capa para la versión de alta capacidad, pero Toshiba y Time Warner querían discos de doble lado, una sola capa. Philips y Sony no creyeron que la gente estuviera dispuesta a voltear los discos, y Time Warner no creía que colocar dos capas en un lado podía funcionar. El resultado: todas las combinaciones; el mercado será quien defina cuáles sobrevivirán.
La tecnología de doble capa tiene una capa reflectora en la parte inferior, con una capa semi-reflectora encima. Dependiendo del lugar en el que se enfoque el láser, rebota de una capa o de la otra. La capa inferior necesita hoyos y áreas lisas ligeramente más grandes para poder leer de manera confiable, por lo que su capacidad es un poco menor que la de la capa superior.
Los discos de doble lado se fabrican tomando dos discos de un solo lado de 0.6 mm y pegándolos de su parte posterior. Para que el grosor de todas las versiones sea el mismo, un disco de un solo lado consiste en un disco de 0.6 mm pegado a un sustrato en blanco (o tal vez en el futuro, uno que consista en 133 minutos de publicidad, con la esperanza de que la gente tenga curiosidad sobre lo que pueda contener). La estructura del disco de doble lado, doble capa se ilustra en la figura 5-24.

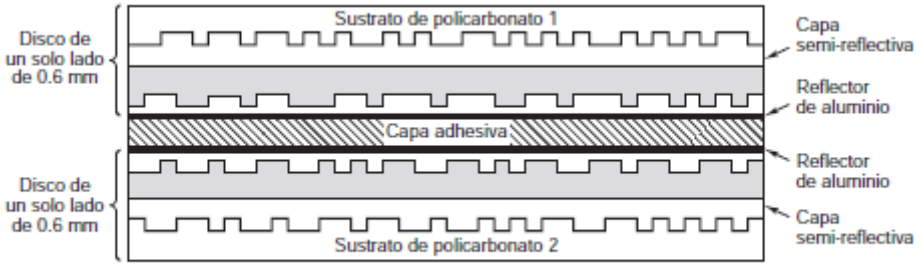


Figura 5-24. Un disco DBD de doble lado y doble capa.

El DVD fue ideado por un consorcio de 10 empresas de aparatos electrónicos para el hogar, siete de ellas japonesas, en estrecha cooperación con los principales estudios de Hollywood (algunos de los cuales son propiedad de las empresas de electrónica japonesas que están en el consorcio). Las industrias de las computadoras y las telecomunicaciones no fueron invitadas al picnic, y el enfoque resultante estuvo en utilizar el DVD para exposiciones de renta y venta de películas. Por ejemplo, las características estándar incluyen la omisión en tiempo real de escenas sucias (para permitir que los padres conviertan una película con clasificación NC17 en una segura para los bebés), sonido de seis canales y soporte para Pan-and-Scan. Esta última característica permite al reproductor del DVD decidir en forma dinámica cómo cortar los bordes izquierdo y derecho de las película (cuya proporción de anchura:altura es 3:2) para adaptarlas a los televisores actuales (cuya proporción de aspecto es 4:3).

Otra cuestión que la industria de las computadoras probablemente no hubiera considerado es una incompatibilidad intencional entre los discos destinados para los Estados Unidos y los discos destinados para Europa, y otros estándares más para los otros continentes. Hollywood exigía esta "característica" debido a que las nuevas películas siempre se estrenan primero en los Estados Unidos y después se envían a Europa, cuando los videos salen en los Estados Unidos. La idea era hacer que las tiendas de video europeas no pudieran comprar videos en los EE.UU. con demasiada anticipación haciendo disminuir las ventas de boletos para las nuevas películas en los cines europeos. Si Hollywood hubiera operado la industria de las computadoras, tendríamos discos flexibles de 3.5 pulgadas en los Estados Unidos y discos flexibles de 9 cm en Europa.

Las personas que idearon los DVDs de un solo/doble lado y una sola/doble capa están ideando nuevos descubrimientos otra vez. La siguiente generación también carece de un solo estándar debido a las disputas políticas por parte de los participantes en la industria. Uno de los nuevos dispositivos es **Blu-ray**, que utiliza un láser de 0.405 micrones (azul) para empaquetar 25 GB en un disco de una sola capa y 50 GB en un disco de doble capa. El otro es **HD DVD**, que utiliza el mismo láser azul pero tiene una capacidad de sólo 15 GB (una sola capa) y 30 GB (doble capa). Esta guerra de los formatos ha dividido a los estudios de películas, los fabricantes de computadoras y las compañías de software. Como resultado de la falta de estandarización, esta generación está des- pegando con bastante lentitud, a medida que los consumidores esperan a que se asiente el polvo para ver qué formato ganará. Esta falta de sensibilidad por parte de la industria nos trae a la men- te el famoso comentario de George Santayana: "Aquellos que no pueden aprender de la historia están destinados a repetirla".

ESTRUCTURA DE DIRECTORIOS EN LINUX

LISTADO DE LAS PRINCIPALES DIRECTORIOS Y SUS FUNCIONES

/

Toda la estructura de directorios en los sistemas basados en UNIX parte de un directorio raíz también llamado directorio root y que se simboliza por una barra inclinada o /. De este directorio, es desde donde nacen todo el resto de directorios, independientemente que estén almacenados físicamente en discos o unidades separadas.
Cualquier dirección de archivo o carpeta en Linux empieza por el directorio raíz o /, seguido de todos los directorios y subdirectorios que que lo contienen, separados cada uno de ellos por /.
A continuación conocerás con más en detalle a todos los directorios principales que parten del directorio raíz, junto con algunos de sus subdirectorios.

/BIN

El directorio /bin es un directorio estático y es donde se almacenan todos los binarios necesarios para garantizar las funciones básicas a nivel de usuario. Solo almacena los ejecutables de usuario, ya que los binarios necesarios para tareas administrativas gestionadas por el usuario root o superusuario del sistema se encuentran en el directorio /sbin.

Incluye también los binarios que permiten la ejecución de varias utilidades estándar de la terminal de Linux, concretamente cat, cd, cp, echo, grep, gzip, kill, ls, mv, rm, ping, su, ps, tar y vi.

/BOOT

Es un directorio estático e incluye todos los ejecutables y archivos que son necesarios en el proceso de arranque del sistema, y que deberán ser utilizados antes que que el kernel empiece a dar las órdenes de ejecución de los diferentes módulos del sistema. Es también donde se encuentra el gestor de arranque GRUB.
En algunas distribuciones, es común que ese directorio se almacene en su propia partición separada del resto. Esto suele darse sobretodo en el caso de de que utilicen LVM por defecto, ya que tradicionalmente el gestor de arranque GRUB (en versiones anteriores a la 2) no podía arrancar desde LVM, por lo que se requería que estuviera en una partición separada.
De hecho, si en una instalación normal de Ubuntu o Debian optas por utilizar LVM, verás que el instalador ya te genera un esquema de particiones con el directorio boot en una partición aparte.
En estos casos es importante prever bien el espacio que le vayas a dar a la partición, ya que a la larga, con la acumulación de diferentes actualizaciones del Kernel, es común que se quede sin espacio y debas hacer limpieza de versiones antiguas del kernel. En este post que te acabo de enlazar lo tienes explicado con más detalle.

/DEV

Este directorio incluye todos los dispositivos de almacenamiento, en forma de archivos, conectados al sistema, es decir, cualquier disco duro conectado, partición, memoria USB, o CDROM conectado al sistema y que el sistema pueda entender como un volumen lógico de almacenamiento.
Siendo esto así, verás que la ruta en la que se encuentra cualquier volumen (partición o dispositivo externo) conectado al sistema siempre empieza por /dev. Este es el directorio que contiene, por decirlo de algún modo, la información de cada uno de los volúmenes, a diferencia del directorio /media, que verás más adelante, que lo que contiene son solo los puntos de montaje, pero no la información real de estos volúmenes.
Para ver esto en la práctica, si abres una ventana de terminal y ejecutas el comando sudo fdisk -l, verás la estructura de particiones de tu sistema. En una instalación típica de cualquier distro GNU/Linux suele ser la siguiente:
/dev/sda1 - Partición principal
/dev/sda2 - Partición extendida
/dev/sda5 - Partición Swap
La partición sda1 suele ser la partición principal, Obviamente si has editado manualmente el esquema de particiones, en tu caso será diferente, esto es solo un ejemplo típico para ayudar a explicar la función del directorio /dev.
Eso en cuanto a particiones. Si se trata de un dispositivo externo, el volumen estará igualmente dentro de /dev, pero en este caso varía el nombre que el sistema le asigna a dicho volumen. Generalmente la estructura suele ser la siguiente (si ejecutas nuevamente el comando sudo fdisl -l con un dispositivo externo conectado lo podrás comprobar tu mismo).
/dev/sdb1
/dev/sdb2
/dev/sdb3
...

/ETC

Es el encargado de almacenar los archivos de configuración tanto a nivel de componentes del sistema operativo en sí, como de los programas y aplicaciones instaladas a posteriori.
Es un directorio que debería contener únicamente archivos de configuración, y no debería contener binarios.

/HOME

Es el directorio de los usuarios estándar, y por lo tanto, el destinado a almacenar todos los archivos del usuario, como documentos, fotos, vídeos, música, plantillas, etc. También incluye archivos temporales de aplicaciones ejecutadas en modo usuario, que sirven para guardar las configuraciones de programas, etc.
Dentro /home hay los directorios personales de todos los usuarios, nombrados según el nombre de usuario utilizado. Así por ejemplo, si en un sistema pongamos que hay dos usuarios denominados User1 y User2, la estructura será así:
/home/User1
/home/User2
Cada directorio de usuario contiene asimismo diferentes carpetas para ayudarlo a clasificar la información. Estas generalmente son: /Documentos, /Imágenes, /Música, /Plantillas y /Vídeos /, así como otros archivos y carpetas ocultas, que son las encargados de guardar la información de configuraciones de las aplicaciones del usuario.
Para visualizar los ficheros ocultos dentro del directorio individual de cada usuario, puedes hacerlo rápidamente mediante la combinación de comandos CTRL + F. Por cierto, y muy importante, todas los archivos y carpetas ocultas en Linux empiezan por un punto, seguido del nombre de la carpeta.
En muchas distribuciones es una práctica recomendada el hecho de ubicar el directorio /home es una partición separada del resto, por tal de facilitar que, en caso de reinstalar el sistema operativo, puedas mantener intacta la partición de la /home, y de este modo mantener todos los archivos personales.

/LIB

Incluye las bibliotecas esenciales que son necesarios para que se puedan ejecutar correctamente todos los binarios que se encuentran en los directorios /bin y /sbin, así como los módulos del propio kernel.
En los sistemas operativos de 64 bits, además de /lib existe otro directorio denominado /lib64, referida a las bibliotecas para aplicaciones de 64 bits.

/MEDIA

Representa el punto de montaje de todos los volúmenes lógicos que se montan temporalmente, ya sean unidades externas USB, otras particiones de disco, etc.
En la mayoría de distribuciones GNU/Linux, desde hace ya algún tiempo, cada vez que se monta una unidad externa, partición, etc., esta se monta dentro del directorio /media y a su vez dentro de un directorio especifico dependiendo del usuario del sistema que monta el volumen.
De este modo, si en un sistema hay varios usuarios, pongamos User1 y User2, los puntos de montaje de los volúmenes que montan cada uno de ellos se mostraran en directorios separados tal como así:
/media/User1
/media/User2

/MNT

Es un directorio vacío que cumple funciones similares a /media, pero que actualmente no se suele utilizar, ya que la mayoría de distribuciones hacen uso de este último para los puntos de montaje temporales.

/OPT

En cierto modo vendría a ser como una extensión del directorio /usr, pero en este caso van todos aquellos archivos de solo lectura que son parte de programas auto-contenidos y que, por lo tanto, no siguen los estándares de almacenar los diferentes archivos dentro de los diferentes subdirectorios de /usr (que sería lo recomendable)
Haciendo una analogía con Windows, vendría a ser algo como el directorio de “Archivos y Programas”, pero en este caso, como hemos dicho, para determinados programas que ya vienen auto-contenidos.

/PROC

Este directorio contiene información de los procesos y aplicaciones que se están ejecutando en un momento determinado en el sistema, pero realmente no guarda nada como tal, ya que lo que almacena son archivos virtuales, por lo que el contenido de este directorio es nulo.
Básicamente son listas de eventos del sistema operativo que se generan en el momento de acceder a ellos, y que no existen dentro del directorio como tales.
En este enlace de LinuxTotal tienes información más detallada sobre las particularidades de este directorio y todo el juego que le puedes sacar a la hora de obtener información muy diversa del sistema.

/ROOT

Vendría a ser como el directorio /home del usuario root o superusuario del sistema. A diferencia de los otros usuarios, que se encuentran todos dentro de /home en sus respectivas subcarpetas, el directorio del usuario root está en su propia carpeta colgando directamente de la raíz del sistema.

/SBIN

Si hemos dicho que en /bin se almacenaban los binarios relativos a las funciones normales de usuario, /sbin hace lo mismo pero para los binarios relativos tareas propias del sistema operativo, y que solamente pueden ser gestionadas por el usuario root, tales como el arranque, tareas de restauración, reparación, etc.

/SRV

Sirve para almacenar archivos y directorios relativos a servidores que puedas tener instalados dentro de tu sistema, ya sea un servidor web www, un servidor FTP, CVS, etc. Así, por ejemplo, en el caso de tener instalado un servidor web, sería buena idea tener el directorio web público dentro de /srv, tal como así:
/srv/www

/SYS

Al igual que /proc, contiene archivos virtuales que proveen información del kernel relativa a eventos del sistema operativo. Es en cierto modo una evolución de /proc, y a diferencia de este último, los archivos se distribuyen de forma jerárquica.

/TMP

Como ya da a entender su nombre, sirve para almacenar archivos temporales de todo tipo, ya sea de elementos del sistema, o también de diferentes aplicaciones a nivel de usuario como puedan ser Firefox o Chrome/Chromium.
Es un directorio dispuesto para almacenar contenido de corta durada, de hecho en la gran mayoría de los casos se suele vaciar de forma automática en cada reinicio del sistema. Aun así, no debes borrar su contenido de forma manual, puesto que puede contener archivos necesarios para ciertos programas o procesos que estén ejecutándose.
Las aplicaciones programadas para almacenar archivos en este directorio deben asumir que solo serán recuperables en la sesión actual. En este sentido, hay otro subdirectorio, /var/tmp, dispuesto igualmente para el almacenamiento de archivos temporales, pero cuyo contenido no se borra de forma automática tras el reinicio del sistema.

/USR

El directorio /usr viene de “User System Resources” y actualmente sirve para almacenar todos los archivos de solo lectura y relativos a las utilidades de usuario, incluyendo todo el software instalado a través de los gestores de paquetes de cada distribución. Contiene los siguientes subdirectorios:
/usr/bin
/usr/include
/usr/lib
/usr/local
/usr/sbin
/usr/share
/usr/src
Antiguamente /usr también contenía la carpeta particular de usuario, junto con todos sus documentos, vídeos, fotos, etc., pero más adelante se creó el directorio /home para este propósito, dejando /usr reservado para los archivos relativos a programas.

/VAR

contiene varios archivos con información del sistema, como archivos de logs, emails de los usuarios del sistema, bases de datos, información almacenada en la caché, información relativa a los paquetes de aplicaciones almacenados en /opt, etc. En cierto modo se podría decir que actúa a modo de registro del sistema.

Cómo montar una partición en Linux

A diferencia de Windows y MS-DOS, en Linux, además de no haber una asignación de letras -a: b: c: d: e:- para las unidades de disco y las particiones, es necesario indicarle al sistema cuando se utilizará una unidad de disco extraible para poder acceder a esta y cuando se dejará de utilizar para poder retirarla y cambiarla por otra. Una vez configuradas las unidades de disco en el sistema se necesitará conocer algunos métodos y atajos para montarlas y desmontarlas rápidamente.

Preparativos para el montaje

Antes de montar la partición, debemos crear una carpeta donde vamos a montarla. Generalmente se suele hacer en /media/ (/mnt en algunas distribuciones), así que es recomendable que la creemos allí:

sudo mkdir /media/lalala
Donde lalala es el nombre que tendrá la carpeta, podemos asignarle el que queramos.
Si el sistema de archivos de la partición que vamos a montar es NTFS se recomienda instalar el controlador ntfs-3g para poder tener soporte de escritura en esa partición:
sudo aptitude install ntfs-3g
Para no tener que cargarlo cada vez que inicie el sistema, podemos editar el archivo /etc/modules:
gksudo gedit /etc/modules

Montaje de la partición

Ahora vamos a montar la partición en la carpeta creada. Esto significa que el contenido de la partición va a aparecer en esa carpeta. El comando para montar discos y/o particiones es mount, y se usa de la siguiente manera:
$ sudo mount -t sistema_archivos [-o opciones] /dev/particion carpeta_montaje
Cabe aclarar que si ocurre algún error durante el montaje, no se pondrán en peligro los datos de la partición, simplemente no será montada.

Parámetros

El significado de los parámetros usados en el comando mount son los siguientes:

- sistema_archivos: es el sistema de archivos de la partición; puede ser vfat (FAT16 y FAT32), ntfs (NTFS) o ufs (UFS y UFS2); hay otros posibles valores, pero no son tratados en este artículo. Si es ufs, entonces debemos indicar de manera obligatoria las opciones ro y ufstype (este último en caso de ser UFS2).
- opciones: son las opciones de montaje, puede tomar más de un valor, en ese caso los valores se separan con comas (,). Algunos posibles valores son defaults (valores por defecto), ro (Read Only, es decir, Solo Lectura) y ufstype (para especificar el tipo de sistema de archivos UFS, en caso de que se use este); si no se especifican opciones especiales, podemos escribir defaults, u obviar este parámetro por completo (quitando también el -o de adelante). En este artículo, usaremos este parámetro únicamente para el montaje de particiones UFS y UFS2, y para el montaje al inicio del sistema (véase el encabezado Montaje al inicio del sistema).
- partición: es el identificador de la partición que vamos a montar; puede ser hdXY en caso de ser un disco IDE o ATA, o sdX,Y en caso de ser SATA; la X es la letra del disco rígido (a para el primero, b para el segundo, etc.) y la Y es el número de partición (1 para la primera, 2 para la

segunda, etc.). Si queremos saber el nombre de las particiones que tenemos en el equipo, basta con ejecutar el siguiente comando:

$ sudo fdisk -l

- carpeta_montaje: es la carpeta donde se montará la partición, es decir, donde aparecerán los datos (archivos y carpetas) de la partición; en la mayoría de los casos se encuentra dentro de /media/, aunque puede estar en cualquier otro lugar (véase el encabezado Preparativos para el montaje).

Ejemplos concretos

Para montar una partición FAT16 o FAT32:
sudo mount -t vfat /dev/particion /media/carpeta_montaje
Para montar una partición NTFS:
sudo mount -t ntfs /dev/particion /media/carpeta_montaje
Para montar una partición UFS:
sudo mount -t ufs -o ro /dev/particion /media/carpeta_montaje
Para montar una partición UFS2:
sudo mount -t ufs -o ro,ufstype=ufs2 /dev/particion /media/carpeta_montaje
Para montar una partición donde está Ubuntu:
sudo mount /dev/particion /media/carpeta_montaje

Montaje al inicio del sistema

Una vez que hayamos conseguido montar la partición, quedará montada mientras el sistema esté en marcha. Cuando reiniciemos o apaguemos el equipo, tendremos que volver a montar la partición. Si queremos que se monte cada vez que iniciamos el sistema, necesitaremos modificar el archivo /etc/fstab:

$ sudo gedit /etc/fstab

Aquí se ha usado gEdit, pero puede usarse cualquier editor de textos, como Nano o Vim. Conviene usar gksudo o kdesu en vez de sudo para iniciar un editor en modo gráfico; el primero es para el escritorio GNOME y el segundo para KDE.

Una vez abierto el archivo, tenemos que cambiar la línea que comience con el identificador de la partición que hemos montado (/dev/hdXY o /dev/sdXY) por la siguiente:

/dev/particion /media/carpeta_montaje sistema_archivos opciones 0 0

Si no existe esa línea, la añadimos al final del archivo.

Los argumentos son los mismos que cuando usamos el comando mount. Aquí, si en opciones no usamos ningún valor, tendremos que escribir defaults, y nos quedaría algo así:

/dev/particion /media/carpeta_montaje sistema_archivos defaults 0 0

Si es una partición FAT16 o FAT32 y no nos funciona con defaults, podemos probar las siguientes opciones:

auto,users,exec,umask=000
defaults,rw,user,auto,umask=0

Con esta última se están dando permisos de lectura, escritura y ejecucion a todos los usuarios. Si queremos restringir estos permisos solo a un grupo particular de usuarios (por ejemplo: users), las opciones deben quedar así:

defaults,rw,user,auto,umask=007,gid=grupo 0 0

Donde grupo debe sustituirse por el grupo de usuarios, por ejemplo, users.

Finalmente, si por cualquier motivo no se detectan bien algunos caracteres (como la letra ñ), debemos añadir la siguiente opción junto con las otras utilizadas, para cambiar el mapa de caracteres:

iocharset=utf8
Por ejemplo, una línea podría quedar así:
/dev/hda0 /media/hda0 vfat defaults,rw,user,auto,iocharset=utf8,umask=000 0 0
Para montar todos los dispositivos listados en el archivo /etc/fstab tenemos que ejecutar el siguiente comando en una terminal:
sudo mount -a
Con esto ya tendemos montada nuestra partición cada vez que se inicie Ubuntu.

Crear un lanzador a la partición

Una vez que tenemos montada la partición, comprobamos que en ocasiones puede resultarnos incómodo acceder siempre a la carpeta /media/hdXY (o a la que hayamos especificado), para solucionar esto podemos hacer dos cosas:

- Crear un lanzador simbólico en nuestra carpeta home o en cualquier otra carpeta
- Crear un ícono en el escritorio que nos dirija a la carpeta de montaje

Lanzador simbólico en /home

La primera opción es crear un lanzador simbólico (son como los accesos directos de Windows), por ejemplo en nuestra carpeta personal home. Con ello conseguiremos acceder a la partición desde esta carpeta y sin duplicar la información.
Escribimos la siguiente línea en la terminal:
$ ln -s /media/carpeta_montaje /home/usuario/carpeta_destino
Donde carpeta_montaje es la carpeta donde montamos la partición, usuario es el nombre de nuestro usuario en el sistema y carpeta_destino es el lanzador simbólico que crearemos para que actúe como carpeta de montaje.

Ícono en el escritorio

Si lo que queremos es crear un ícono en el escritorio, debemos hacer lo siguiente:

En GNOME

1. Desplegamos el menú contextual del escritorio (botón derecho del ratón)
2. Seleccionamos la opción Crear enlace
3. Escogemos el tipo Enlace
4. Rellenamos los campos en blanco con los datos requeridos
5. Aceptamos para que se cree el enlace

En KDE

1. Desplegamos el menú contextual del escritorio (botón derecho del ratón)
2. Seleccionamos la opción Crear nuevo -> Enlace a dispositivo -> Disco duro
3. En la pestaña General, escribimos el nombre del ícono
4. En la pestaña Dispositivo, escribimos el identificador de la partición (/dev/hdXY)
5. Aceptamos para que se cree el enlace

Desmontaje de la partición

Si por cualquier motivo deseas desmontar la partición, no tienes más que escribir esto en terminal:

sudo umount carpeta

Donde carpeta es la ubicación de la carpeta donde está montada la partición (por ejemplo: /media/hdaX).